*i*MASON: Towards Influence-driven Multi-level Analysis of Online Social Networks

Li Hui

School of Computer Engineering

A thesis submitted to the Nanyang Technological University
in fulfilment of the requirement for the degree of
Doctor of Philosophy

2012

Acknowledgments

I wish to render my sincere appreciation to my supervisor, **Associate Professor Sourav S Bhowmick**, for the invaluable advice and guidance. Thanks to him for bring me into the world of research. I have learned so much from him about how to do a good research, write technical papers and give presentations. Without his consistent support and help, the research would not have been so enriching and fulfilling.

I'd like to thank **Assistant Professor Sun Aixin** for helpful discussion and suggestions in my research work during the last four years. I would also thank Dr. Lim Ee-Peng, Dr. Zhang Jun and Dr. Anwitaman Datta for their advice and suggests. I am also grateful to Dr. Erwin Leonardi for all his help in my research.

I am grateful to all my fellow friends in Centre for Advanced Information Systems (CAIS): Ang Hock Hee, Ardian, Chua Huey Eng, Li Guangxia, Li Chenliang, Li Bin, Liu Xin, Seah Boon Siew, Sun Hongyang, Wu Min, Wu Pengcheng, Xia Hao, Zhao Peilin, Zhu Linhong and Zhuang Jinfeng. Also thanks to Dai Hanbo at Singapore Management University and Cao Yangjie at Xian Jiaotong University.

Many thanks to Mr. Zhou Yong, Mr. Jin Changjiu and Mr. Fajar for all their supports for me.

Thanks to School of Computer Engineering and Nanyang Technological University for the research scholarship. Also thanks to Mr. Lai Chee Keong and Mr. Chua Chiew Song in Centre for Advanced Information Systems for their technical support and the facilities.

Last but not the least, I would like to thank my parents, for their love, support and patience throughout these years. I would not have come this far without their constant encouragement.

Contents

List of Figures

List of Tables

Abstract

Social network is a social structure of nodes that are tied by various kinds of relationships, such as kinships, friends, web links, colleagues, citation links, etc. Recently, large on-line social networks have become very popular among web users. A key feature of many online social networks is that they are driven by *social influence* between users. Specifically, the structural and semantic properties of individuals, communities and network can be traced back to the social influence effect. Hence, studying the social influence between users in a network is of great importance in many applications such as viral marketing, online advertising, advanced web architecture design, etc.

In this dissertation, we propose a framework called *i*MASON (**i**nfluence-driven **M**ulti-level **A**nalysis of online **SO**cial **N**etworks) that systematically investigates social influence effect in social networks from multiple granularity including *individual-level*, *community-level* and *network-level*. Specifically, arrays of novel influence-related problems are tackled and many new interesting findings are uncovered in this thesis. We propose four different models under *i*MASON in order to analyze the social influence effect at different levels. At individual level, we propose to study and predict *blog cascade affinity*. We also propose a novel algorithm called CASINO to evaluate each node's *influence* and *conformity*. At community level, we propose *AffRank* to rank the ability of product communities to absorb new members which is driven by social influence effect. At network level, we propose a novel algorithm called PATINA to solve *influence maximization* problem. Moreover, we also propose a novel *cascade model* called C^2 (Conformity-aware Cascade) model taking into account conformity of nodes in influence cascade process. We propose CINEMA to solve the influence maximization problem under C^2 model. We test *i*MASON framework in many different real-world social networks including Blogosphere, Twitter, Blippr, Epinions, Slashdot, etc. Our experimental results show that *i*MASON exhibit superior performance compared to state-of-the-art techniques in all these three levels of social influence study.

Chapter 1

Introduction

Social network analysis (SNA) views social relationships in terms of network theory consisting of nodes and ties. Nodes are the individual actors within the networks; ties are the relationships between the actors. The relationship could be friendship, communication, trust, etc. In the sequel, we will use the term *node* and *individual* interchangeably. The power of SNA stems from its difference from traditional social scientific studies, which assume that it is the attributes of individual actors that matters. SNA produces an alternate view, where the attributes of individuals are less important than their relationships and ties with other actors within the network. The reason is that these relationships and ties are driven by *social influence* [2, 3] which is the most important phenomenon that distinguishes social network from other networks [4].

1.1 Background

In social psychology, social influence occurs when an individual's thoughts, feelings or actions are affected by other people [2, 3]. Majority of social ties in social networks such as who-believe-whom, who-emails-whom, who-likes-whom or who-borrowed money from-whom can be concluded as social influence effect. We refer to these networks which are driven by social influence effect as *influence-driven social networks*. In this thesis, we focus on these networks and study how the social influence phenomenon affects them. For example, Figure 1.1 depicts a conversation between users u_1 and v which is a common scenario in many kinds of social

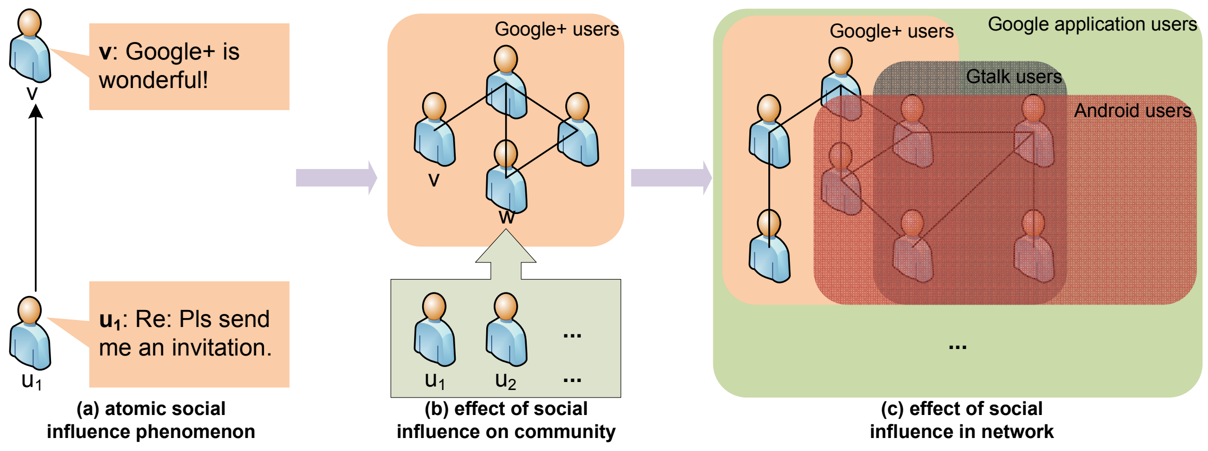

v: Google+ is wonderful!
v
u1: Re: Pls send me an invitation.
u1
(a) atomic social influence phenomenon
Google+ users
v
w
u1
u2
...
(b) effect of social influence on community
Google+ users
Google application users
Gtalk users
Android users
...
(c) effect of social influence in network

Figure 1.1: The effect of social influence phenomenon in social network.

networks [3, 5, 6, 7, 8, 9, 10]. Suppose that v recommends a new application of *Google+* to her neighbors. One of her friend u_1 sees the recommendation and decides to have a try. In summary, u_1 is influenced by v to use a new application. Similarly, u_2 is also influenced by w to use this application. Such effects will result in a growth of the *Google+* user community which is shown in Figure 1.1(b). Thus, the evolution of a community can be traced back to the effect of social influence. Similar to the evolution of *Google+* community, *Gtalk* user and *Android* user communities also evolve in this way (*i.e.,* Figure 1.1(c)). Such changes will result in the evolution of the whole *Google* user network. It is obvious that most of the changes and evolutions of social networks can be seen as the result of the atomic social influence effect in Figure 1.1(a). Thus, in this thesis we propose a framework called *i*MASON (*influence-driven Multi-level Analysis of SOcial Networks*) to investigate SNA from social influence effect in all the three levels of study. It includes individual-level which investigates social influence effect between individual users, community-level which investigates social influence in community evolution, and network-level which investigates social influence propagation within the whole network.

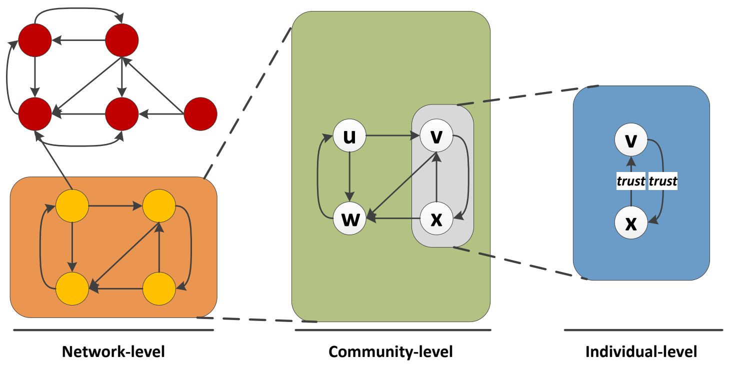


Figure 1.2: Different levels of social influence analysis.

1.2 Overview of Existing Research

Generally speaking, there are three levels of research directions in SNA, namely, *individual* analysis, *community* analysis and *network* analysis, which focus on studying the individual behavior, community evolution, and global network properties, respectively. Thus, the social influence study can also be classified into these three levels which is shown in Figure 1.2. In the following, we describe the key features of social influence and its relationship with the evolution of social network at these three different levels.

Individual-level. Individual-level social influence happens in person-to-person form [11]. It can be referred to as atomic social influence phenomenon [12]. As shown in Figure 1.1(a), there are obviously three factors: the *subject node* who is influencing others (*i.e.,* v), the *social tie* where influence flows (*i.e.,* $\overrightarrow{u_1 v}$) and the *object node* who is influenced (*i.e.,* u_1).

One key feature with respect to the *subject node* is the *influence* of it. In many other networks, there exist ways to measure the importance of each node (*i.e.,* node centrality in general graph [13, 14, 15, 16, 17], *PageRank* in World Wide Web [18]). Similarly, nodes in a social network also exhibit different abilities to influence others. As the connections between nodes in social networks are driven by social influence effect, existing approaches of measuring nodes' importance in other networks may not be applicable in social networks. Moreover, different social networks may be driven by different types of influence (*i.e.,* friendship, trust, cooperation, etc.). Thus, the influence mining algorithm in different social networks may also vary from each other. For example, many models have been proposed to find the most influential blogger [19, 20], expert in a forum [21], the most influential expert in a coauthor network [22], etc.

Each *social tie* in social networks may carry different *weight*. The measure of *weight* may vary in different kind of social networks. For example, degree of trust in *Epinions* has been studied in [23], polarity of the communication have been studied in [24, 25, 26], presence of the communication have been predicted in [27].

Community-level. As an effect of the social influence, people in a social network with similar hobbies, job and education background may form communities [28, 29, 30, 31]. These communities in social network are highly dynamic with new members joining, old members leaving, etc. Thus, detecting communities within a highly dynamic social network is a challenging task. Researchers have proposed several methods in order to find communities within different social networks using either structural information or a combination of structural and content information. Authors in [29, 32] have proposed methods using structural information to extract communities within social networks. Besides, works in [30, 33, 34, 35, 36, 37] have explored ways of using both structural information and social influence effect in discovering communities within social networks. Moreover, many models have been proposed to investigate how social influence affects the evolution of a given a community [1, 38].

Network-level. Social influence effect have made social network significantly different from others especially in that social networks exhibit much higher clustering coefficient than other networks. Watts and Strogatz [39] found a clustering coefficient of 0.79 for the actor network where two actors are linked if they appear in the same movie. However, a corresponding power grid network or random graph with the same size has a coefficient of 0.08 or 0.00027, respectively. It indicates that the nodes in social networks are highly clustered compared to other networks. Such a property makes social network an important media to spread information across a group of people [5, 40, 41]. Thus, how to maximize the *information spread* from a limited set of *seed nodes* have been proposed as a problem in this field. It is well known as *influence maximization* problem [4, 42, 43]. Kempe et al. [4] has proved that the problem is NP-hard and submodular. Thus, greedy-based approaches can achieve within 63% of the optimal solution. Hence, many greedy-based algorithms have been proposed to solve this problem [42, 43]. However, all these approaches are computationally expensive. To address this problem, a series of heuristic approaches such as *DegreeDiscountIC* [42] and LDAG [44] have been proposed. However, these approaches exhibit inferior result quality compared to those greedy approaches. In summary, these heuristic approaches sacrifice seed quality for running time.

1.3 Motivation

In the preceding sections, we highlighted the effect of social influence in social network study and discussed issues with respect to social influence research at individual-level, community-level and network-level. In this section, we discuss the limitations of existing social influence study and the motivation of this thesis.

Individual-level. As discussed in Section 1.2, existing works on individual-level social influence study have either focused on the *subject node* (*i.e.,* v in Figure 1.1) which is the node influencing others or the communication (*i.e.,* $\overrightarrow{u_1 v}$) which is the link that influence flows along. However, these works have ignored the *object node* (*i.e.,* u_1) which is the node being influenced by others. The works in influence mining have showed that different nodes may exhibit different ability to influence others. Similarly, different nodes may also exhibit different inclination to be influenced by others. For example, in Figure 1.1(b) u_1 and u_2 may exhibit different inclination to be influenced. Thus, it is possible that u_1 joins *Google+* network but u_2 does not

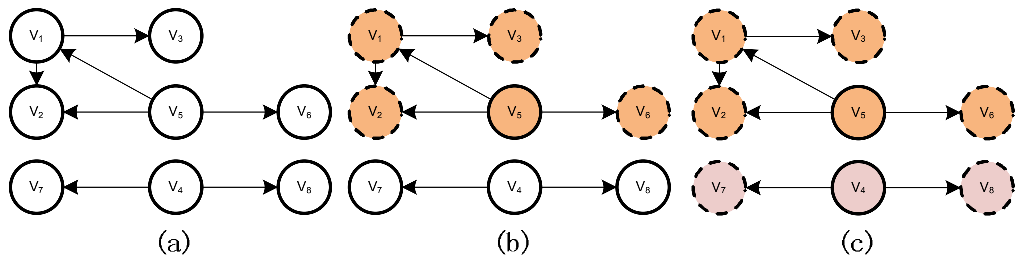


Figure 1.3: (a)Influence maximization scenario. (b)Influence of node v_5. (c)Influence of node v_5 and v_4.

as he is hard to be convinced. *Can we predict the probability of u_1 and u_2 to join a community by analyzing the information embedded in the network?* In this thesis we provide an affirmative answer to this question by proposing a systematical model to predict the probability of a blogger to join a blog cascade.

Besides, influence spread from v to u_1 in Figure 1.1 not only depends on the influence of v but also depends on the inclination of u_1 to be influenced. However, *none of existing work has evaluated the inclination of u_1 to be influenced.* In this thesis, we propose a novel algorithm called CASINO in Chapter 4 to address this problem. It quantifies the influence and conformity of each individual in a network by utilizing the *positive* (*e.g.,* trust, agreement) and *negative* (distrust, disagreement) relationships between individuals.

Community-level. Within the network of *Google* users in Figure 1.1(c), there are different user communities such as *Google+*, *Android* and *Gtalk*. As the network is evolving, these communities may grow at different speed. *An important question in viral marketing is that, among these communities which one is more probable to absorb new members or grow faster?* The answer to this question is important to marketing and business strategists [31, 45]. However, existing works in community-level social influence study have not addressed this problem. In this thesis, we address this problem by proposing a novel model *AffRank* to rank product communities according to their ability to absorb new members in future.

Network-level. As discussed in Section 1.2, social network has become an important media to spread information across a group of people. Suppose a company wants to market a new product through the social network of potential customers as depicted in Figure 1.3. This network consists of eight individuals represented by nodes and the edges between them represent connections or relationship between individuals. The company has a limited budget such that it can only select a small number of initial users (significantly less than 8 in Figure 1.3) in the network to use it for free. It hopes that these users will like the product and trigger a cascade of social influence by recommending the product to other friends, and many individuals will ultimately adopt it. Many cascade models such as *independent cascade* model (IC), *weighted cascade* model (WC) and *linear threshold* model (LT) have been proposed to describe the way how influence propagate from person to person [4]. Assume influence propagates within Figure 1.3 according to IC model with $p = 1$, thus influence propagates along each edge with probability equal to 1. If the company expects to select only one initial user they may select v_5 first as it may influence the most number of potential users which is shown in Figure 1.3(b). If the number of initial users is two, the company may select v_5 first and then v_4 as shown in Figure 1.3(c). Obviously, the company expects the size of the influenced population to be as large as possible. Such a problem is formally defined as *influence maximization* problem [4] and has drawn much attention recently [42, 43].

A key limitation of the existing approaches is their efficiency and scalability. It requires more than 14 hours for a desktop with 1.86GHz Due-Core CPU and 4GB RAM to find seed set of size 100 in a network with only 37,154 nodes according to the research in [42]. Given the existence of real-world networks containing millions of nodes, it will take days by these greedy approaches to find seeds (see in Chapter 6). Consequently, deployment of these techniques on large-scale social networks is not feasible. To alleviate the aforementioned performance bottleneck, Chen et al. [42] proposed several heuristics that exploit degree information to bring in tremendous improvement to the computation time. Specifically, these heuristic-based techniques are orders of magnitude faster than all greedy algorithms. However, the quality of seed set with respect to influence spread can be inferior compared to greedy approaches for many networks. *Is it possible to design a greedy approach that can significantly reduce the computation time for seed set without compromising on the quality of influence spread?* We provide an affirmative answer to this question in this thesis by proposing a novel algorithm PATINA which solves the problem in a parallel and distributed way. It first partitions the network into a set of non-overlapping subnetworks. Each subnetwork is associated with a COG-*sublist* which stores the *marginal gains* of the nodes in the subnetwork in descending order. The node with maximum marginal gain in each COG-sublist is stored in a data structure called MAG-*list*. These structures are manipulated by PATINA to efficiently find the seed set.

Moreover, taking into account the conformity of nodes we proposed a novel *Conformity-aware Cascade model* (C^2). According to the model, the influence propagation probability from node u to v not only depends on the influence of u but also depends on the conformity of v. We optimize the *seed selection* phase in PATINA such that it computes each node's influence size according to C^2 model instead of IC or WC in each iteration.

1.4 Overview of Research

In order to address the aforementioned limitations, we propose in this thesis a framework called *i*MASON (influence-driven Multilevel Analysis of SOcial Networks) which systematically studies large online social networks from the perspective of social influence. In this thesis, we investigate social influence phenomenon at individual-level, community-level and network-level over many different real-world social networks (*i.e.,* Blogosphere, Twitter, Social rating networks, Paper collaboration and Wiki).

The architecture of *i*MASON framework is shown in Figure 1.4 which depicts all the components that constitute the framework. The framework takes real-world social network data as input. Given an arbitrary real-world social network, *i*MASON investigates the social influence effect at three different levels: individual, community and network. Four different models are

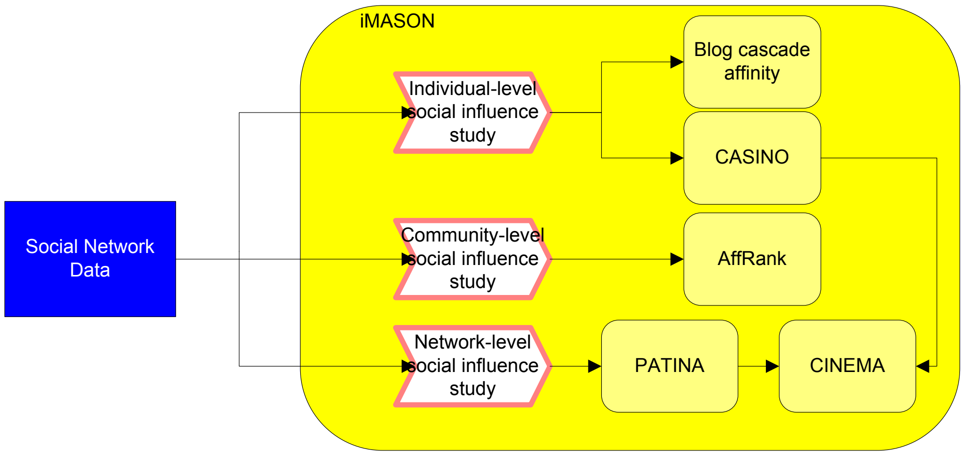

iMASON
Social Network Data
Individual-level social influence study
Community-level social influence study
Network-level social influence study
Blog cascade affinity
CASINO
AffRank
PATINA
CINEMA

Figure 1.4: The *i*MASON framework.

Table 1.1: Multi-level models with references to the chapters in this thesis.

Level	*Models under i*MASON	*Objective*
Node	**Blog Cascade Affinity (Chapter 3)**	discover cascade joining behavior of bloggers
	CASINO (Chapter 4)	evaluate the influence of each individual incorporating the conformity factor
Community	***AffRank* (Chapter 5)**	predict the ability of communities to absorb new members which is actually driven by social influence effect
Network	**PATINA (Chapter 6)**	solve influence maximization in large social networks
	CINEMA (Chapter 7)	solve influence maximization with respect to conformity of nodes

proposed under *i*MASON in order to address the limitations of existing work in aforementioned social influence study. Firstly, *blog cascade affinity* is investigated at individual-level. CASINO is proposed to evaluate the influence and conformity of individual nodes. Secondly, *AffRank* model is proposed to rank the product communities according to their ability to absorb new members. Thirdly, at network-level a novel algorithm PATINA is proposed to maximize influence propagation within large networks. Moreover, we propose CINEMA by optimizing PATINA with respect to *conformity*. All these models are summarized in Table 1.1.

We proposed four models that aim to address limitations discussed in Section 1.3 at all the three levels under *i*MASON framework.

- At node level, we modeled a blog's inclination to join a specific cascade as *cascade affinity*. As a piece of information spreads over the underlying blogosphere via hyperlinks, it creates a trace, called a blog cascade [46]. Nodes in the cascade are the posts in blogosphere that are correlated with some specific information and edges, which represent influence relations, are hyperlinks between the posts. We investigated the affinity from seven different aspects: *elapsed time*, *number of participants*, *star-likeness ratio*, *number of friends*, *popularity of participants*, *citing factor*, and *initiator-media links*. We proposed both probabilistic and non-probabilistic learning models and predicted the *blog cascade affinity* for an arbitrary pair of cascade and blog. Besides, we investigated the social influence process in detail and find an important aspect which have been ignored in previous influence mining models: *conformity*. It describes a person's inclination to be influenced by others. We proposed a novel algorithm CASINO which takes into account both influence and conformity factors and finally assigns each person a pair of indices indicating his influence and conformity.

- At community level, we studied product communities' abilities to attract new customers from the aspects of several features such as *affinity rank history*, *average ratings*, and *affinity evolution distance* which have not been proposed before. We modeled the evolution of product community ranks as a regression problem and proposed an *AffRank* model to predict the future *affinity rank* of a product community. To the best of our knowledge, we are the first to study the affinity rank in social rating networks.

- At network level, we proposed an algorithm called PATINA which effectively reduces both space and time cost for solving influence maximization problem. Besides, it can be easily adopted in distributed environment and will thus accelerate the computation much more. To the best of our knowledge, we are the first to propose a solution that can be easily scaled to distributed environment and run in parallel. Moreover, we take into account the conformity of nodes in influence propagation and propose a novel *Conformity-aware Cascade model* (C^2) based on our study of *conformity* in CASINO model. Our study revealed that state-of-the-art algorithms cannot provide satisfactory result if we take into account the conformity of nodes. Thus, we propose a conformity-aware algorithm called CINEMA in order to solve the influence maximization problem in C^2 model.

1.5 Novel Contributions

The *i*MASON framework investigates social networks from the social influence perspective at three different levels, namely, *individual*, *community* and *network*. To study social influence effect at all these three levels, we propose several novel contributions as followings.

- **A set of novel features to describe the personal behavior and community evolution:** A group of content-oblivious macroscopic and microscopic features have been proposed to measure the blog cascade affinity. To the best of our knowledge, these features have not been studied together in the context of blogosphere. Moreover, the features *star-likeness ratio*, *citing factor*, and *initiator-media links* have not been studied in any previous work. On the other hand, to measure the evolution of product communities in social rating networks we have proposed features such as *affinity rank history*, *affinity evolution distance*, and *average rating*. They are shown to exert significant influence on affinity rank prediction in our study.

- **A novel perspective to investigate social influence phenomenon:** To measure the influence of each individual node, we propose to study the social influence phenomenon by taking into account the factor of *conformity*. We are inspired by the concept of *conformity* in social psychology research [2] and propose to evaluate it in real-world social networks. To the best of our knowledge, we are the first to study the interplay of influentials and conformers with the goal of social influence analysis.

- **A set of algorithms and models for extracting novel information:** We propose several models and algorithms to study social network from the aspect of social influence. With respect to individual-level study, we propose a blog cascade affinity model using a series of novel features to predict a blogger's inclination to join a given cascade. We also propose a novel algorithm CASINO to evaluate the influence and conformity of nodes in an arbitrary social network. With respect to community level SNA, we propose an *AffRank* model using a group of novel features to predict the future affinity rank of a product. With respect to network level SNA, we propose a novel algorithm PATINA to maximize influence propagation within large social networks. Moreover, we propose a conformity-aware algorithm called CINEMA which optimizes PATINA with respect to conformity.

1.6 Organization of the Thesis

The rest of this thesis is organized as follows.

- In Chapter 2, we discuss existing works in the area of social influence study.
- In Chapter 3 and Chapter 5, we present our work on node-level social influence study. Firstly, in Chapter 3, we reveal the problem of blog cascade affinity. In this work, we investigate each individual blog's probability of joining a given cascade based on several proposed features. Besides, in Chapter 4 we propose a conformity-aware social influence analysis model CASINO which aims to find influentials as well as those people who are most easily to be influenced.

- In Chapter 5, we discuss our work on community level social influence of *AffRank* which ranks the communities according to their abilities to absorb new members in future. Studying the affinity ranks of product communities will help us to predict a community's ability of attracting new members.

- In Chapter 6 and Chapter 7, we investigate the network level social influence study. Firstly, in Chapter 6 we present a model PATINA, which can run in both single node mode and distributed environment, to solve the influence maximization problem in large real-world social networks. Secondly, in Chapter 7 we propose a novel cascade model called *Conformity-aware Cascade* model (C^2) by taking into account the conformity of nodes. In order to solve the influence maximization problem in C^2 model, we proposed CINEMA by optimizing PATINA with respect to *conformity*.

- In Chapter 8, several improvements and extensions under current consideration are discussed, which constitute to a part of the future work of this research.

Chapter 2

Literature Review

In this chapter, we present an overview of the representative research work in social influence study. Those work can be classified into three levels. Figure 2.1 shows the categorization of social influence study in detail. We shall follow this structure and discuss each of them. After that, we summarize these work and compare them with ours.

2.1 Individual-level Analysis

Recall from Chapter 1, there are three main research areas with respect to individual-level social influence study: the *subject node* which influence others, the *social tie* where influence flows and the *object node* which is influenced. In the sequel, we discuss each of them in sequence.

2.1.1 Subject Node

Influential mining problem [47, 48, 49] is the focus in the research field with respect to *subject node*. It aims to answer the following question: given a social network, which are the most important nodes with respect to a specific application? In the following, we review some representative work in this field.

Out-break. Leskovec et al. [43] discussed the problem of out-break detection in networks. The authors answered the following question: which blogs should we read to avoid missing

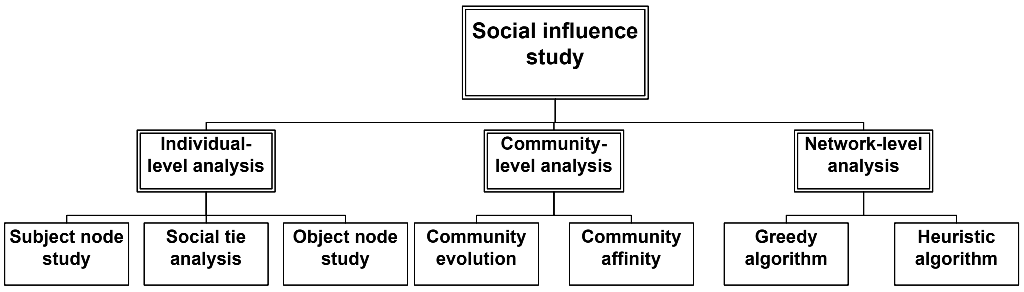


Figure 2.1: Social influence study categorization.

important stories? In the paper, the authors proposed a model to find a set of nodes as sensors so that once outbreak happens the system sensors can detect the outbreak as soon as possible. Nodes selected from the model can be referred to as a *placement* of sensors. According to their approach, the problem is formed as:

$$\max_{A \subseteq \Lambda} R(A) \;\; \textit{subject to} \;\; c(A) \leq B \tag{Eq. 2.1}$$

where R(A) is a *placement score* of a placement A to be maximized, c(A) is the related cost of such a placement A, and B is a given budget. To accomplish the task, they introduced a *penalty function*:

$$\pi(A) = \sum_i P(i)\pi_i(T(i,A)) \tag{Eq. 2.2}$$

where for a placement $A \subseteq \Lambda$, $T(i,A) = \min_{s \in A}(i,s)$ is the time until event i is detected by one of the sensors in A, P is a given probability distribution over the events, $\pi_i(t)$ denotes the penalty of detecting event i at time t. $T(i,\varnothing)$ is set to ∞, $\pi_i(\infty)$ is set to some maximum penalty incurred for not detecting the event i. Then R can be defined as:

$$R(A) := \sum_i P(i)R_i(A) = \pi(\varnothing) - \pi(A). \tag{Eq. 2.3}$$

Following this model, the authors showed by experiment that they can find a series of important nodes in blogosphere such that detecting information cascades with the minimum cost is possible.

Influential blogger. Agarwal et al. [20] proposed a model to measure the significance of blogs. They developed a ranking algorithm to discover influential bloggers with the help of a *post influence* graph where the influence of a blog post flows along the post-post links. If I denotes the influence of a node (or a blog post p), then *InfluenceFlow* across that node is:

$$\mathit{InfluenceFlow}(p) = \omega_{in} \sum_{m=1}^{|\iota|} I(p_m) - \omega_{out} \sum_{n=1}^{|\theta|} I(p_n) \qquad \text{(Eq. 2.4)}$$

where ω_{in} and ω_{out} are the weights that can be used to adjust the contribution of incoming and outgoing influence, respectively. p_m denotes all the blog posts that link to p, where $1 \leq m \leq |\iota|$; p_n denotes all the blog posts that are referred by p, where $1 \leq n \leq |\theta|$; $|\iota|$ and $|\theta|$ are the total number of in-links and out-links of p. *InfluenceFlow* accounts for the part of a post's influence that comes from in-links and out-links. The overall influence of a blog post p can be defined as:

$$I(p) = \omega(\lambda) \times (\omega_{com}\gamma_p + \mathit{InfluenceFlow}(p)) \qquad \text{(Eq. 2.5)}$$

where ω is a weight function which rewards or penalizes the influence score of a blog post depending on the length λ of the post. ω_{com} denotes the weight that can be used to regulate the contribution of the number of comments γ_p towards the influence of blog post p. Hence, for a blogger B, the influence score of each of B's N posts can be calculated as the blogger's *iIndex*:

$$\mathit{iIndex}(B) = \max(I(p_i)) \qquad \text{(Eq. 2.6)}$$

where $1 \leq i \leq N$. Thus, bloggers within a blogsphere can be ranked according to *iIndex*.

Heat diffusion. Ma et al. [50] used heat diffusion models to find a set of k influential candidates as target for marketing strategy in social networks. Particularly, the influence propagation is modeled as a heat diffusion process within social networks where the influence a node i receives at a particular timepoint t follows a heat diffusion formula as the following.

$$\frac{f_i(t+\Delta t) - f_i(t)}{\Delta t} = \alpha(-\tau_i f_i(t) + \sum_{j:(v_j,v_i)\in E} \frac{1}{d_j} f_j(t)) \qquad \text{(Eq. 2.7)}$$

In the above equation, τ_i is a flag to identify whether node i has any outlinks, such that $\tau_i = 0$ if node i does not have any outlinks, otherwise, $\tau_i = 1$. Solving the equation, the influence that nodes receive at timepoint t can be expressed as the following.

$$\mathbf{f}(t) = e^{\alpha t H}\mathbf{f}(0), H_{ij} = \begin{cases} 1/d_j, & (v_j, v_i) \in E, \\ -\tau_i, & i = j, \\ 0, & otherwise \end{cases} \tag{Eq. 2.8}$$

Based on this idea, the top-k candidates whose heat diffused to the largest scope are selected using a greedy algorithm.

Twitter authority. In a different media, Pal et al. [51] used the count of original tweets, conversational tweets, and re-tweets of a tweeter as features to rank the *authority* of each tweeter in the context of different topics. They employed a *Gaussian Mixture Model* to compute the authority score of each tweeter. Formally, the authority score for twitter i can be computed as the following.

$$R_G(x_i) = \prod_{f=1}^{d} [\int_{-\infty}^{x_i^f} N(x; \mu_f, \sigma_f)]^{w_f} \tag{Eq. 2.9}$$

In the above equation, w_f is the weight that is put on feature f; x_i^f is the associated value of node i on feature f; $N(x; \mu_f, \sigma_f)$ is the univariate Gaussian distribution with model parameters as μ_f and σ_f. The authority score defined above helps in devising a total ordering under "$\leq$" over all the users. To validate their results, they conducted a survey to rate the authority of the tweeters and use it as the ground truth for authority ranking.

The comparison of all the aforementioned work is summarized in Table 2.1. They did not address the following issues. First of all, there exist positive edges and negative edges in *Twitter* where positive edges represent agreement while negative ones represent disagreement relationships. The aforementioned models failed to distinguish negative edges from positive ones. They treated both kinds of edges equally. Secondly, these existing researches lack justification over the influential mining result. In contrast, our work in Chapter 4 addressed both of the above issues in that we not only take into account both positive and negative edges but also justify the mining result in link prediction task.

Table 2.1: Influential mining models comparison

	Negative edge	Edge type	Text analysis	Justification
Out-break [43]	X	Undirected	X	X
Influential blogger [20]	X	Directed	✓	X
Heat diffusion [50]	X	Directed	X	X
Twitter authority [51]	X	Directed	✓	user survey
CASINO	✓	Directed	✓	link prediction

2.1.2 Social Tie

A common problem that is related with *social tie* is link mining which can be described as follows: given a pair of unconnected nodes i, j, what is the probability that they are connected in a future time t'. Since social networks consist of individuals, the links between the individuals tend to mirror or, in some cases, establish new information propagation kernels. Studying the *social tie* allows discovery and usage of information dissemination within social networks. We review some representative link mining work in the following and then compare them with our work.

Common neighbor. Liben-Nowell et al. have proposed an algorithm to solve the problem of link prediction [27]. They developed approaches to link prediction based on measures of the "proximity" of nodes in a network. Experiments on large social networks suggest that information about future interactions can be extracted from network topology alone. Their approach is based on the idea that two nodes x and y are more likely to form a link if $\Gamma(x)$ and $\Gamma(y)$ have large overlap where $\Gamma(x)$ denotes the neighbors of x. It follows the natural intuition that such node pairs represent authors with many colleagues in common, and hence are more likely to come into contact. Thus, the authors set several different score measures to evaluate the probability x and y will cooperate in future.

$$score(x,y) := |\Gamma(x) \cap \Gamma(y)| \qquad \text{(Eq. 2.10)}$$

$$score(x,y) := |\Gamma(x) \cap \Gamma(y)| / |\Gamma(x) \cup \Gamma(y)| \qquad \text{(Eq. 2.11)}$$

$$score(x,y) := |\Gamma(x)| \cdot |\Gamma(y)| \qquad \text{(Eq. 2.12)}$$

Equation Eq. 2.10 is the concept of common neighbors, Newman [52] has verified a correlation between the number of common neighbors of x and y and the probability that they will collaborate in the future. Equation Eq. 2.11 utilized *jaccard coefficient* which is widely used in information retrieval, measuring the similarity of features between x and y. Equation Eq. 2.12 is based on *preferential attachment* model [53, 54] which claimed that the probability a new edge involves node x is proportional to $\Gamma(x)$. The authors also proposed that the probability of connecting x and y is correlated with the product of the number of collaborators of x and y.

Historical study. O'Madadhain et al. [55] proposed an algorithm for prediction and ranking of link existence based on event-based network data. The main contribution of their paper is predicting the probability for pair of individuals to co-participate in the same event. The problem can be formally described as follows: "given the information of a series of events, will entities v_j and v_k co-participate in at least one event in a future specified interval?". The authors treat this task as a data-driven classification problem (in which "co-participating" is one class, and "not co-participating" is another). The methods used are primarily probabilistic classifiers, which assigns a probability to each class conditioned on the values of a set of specified features, whose nature may vary depending on the data set. They defined the conditional probability as the following:

$$p(v_j, v_k \in P_{t,t+\Delta t} | f(\Upsilon_{1,t}, X, Y) = w) \qquad \text{(Eq. 2.13)}$$

where $v_j, v_k \in P_{t,t+\Delta t}$ is a binary proposition defining whether entities v_j and v_k co-participate in any event during the period $t, t+\Delta t$, f is a function returning a vector w of feature values,

$\Upsilon_{1,t}$ is the historical event data up to time t, and X,Y are the relevant entities and event covariate data. By computing the conditional probability as above, they are able to predict the probability that a pair of individuals will co-participate in a event.

Signed edge prediction. In many social networks, there exists different attitude attached to the edges. For example, in *Epinions* the social ties between users may express trust or distrust; in *Slashdot* the social tie may indicate agreement or disagreement. The edges attached with trust/agreement can be labeled as *positive*; the edges representing distrust/disagreement can be labeled as *negative*. Leskovec et al. [24] investigated some of the underlying mechanisms that determine the signs of links in large social networks where interactions can be both positive and negative. They used logistic regression to predict the signs of edges in signed networks by exploiting a series of features as following.

$$P(+|x) = \frac{1}{1+e^{-(b_0+\Sigma_i^n b_i x_i)}} \qquad \text{(Eq. 2.14)}$$

Within the above formula, x is a vector consisting of features $(x_1,\ldots,x_n)$ and $b_0,\ldots,b_n$ are the coefficients learned from the training data. All these features are solely extracted from the structure of the network. They showed that their model significantly improves previous approaches.

OOLAM. Cai et al. [56] proposed another algorithm called OOLAM (an Opinion Oriented Link Analysis Model) by introducing a new feature (*i.e.,* influence) aside from the 7-dimensional degree features in [24]. A PageRank-like algorithm was developed to compute the influence of individual users and then use it as another feature in an SVM classifier to predict the signs of edges. They showed that by taking into account the social influence of individual users the accuracy of edge sign prediction can be significantly improved. Based on this, they categorized influence personae into *Positive Persona*, *Negative Persona*, and *Controversy Persona*. *Positive* and *Negative Personae* represent users with high positive and negative influence, respectively. The last kind of *Controversy Persona* represents a group of individuals who are liable to be challenged or supported by many.

Table 2.2: Link mining models comparison

	Negative edge	Edge type	Social influence features	Object node
Common neighbor [27]	X	Undirected	X	X
Historical study [55]	X	Undirected	X	X
Signed edge prediction [24]	✓	Directed	X	X
OOLAM [56]	✓	Directed	node influence	X
CASINO	✓	Directed	node influence & conformity	✓
Blog cascade affinity	X	Directed	#friends, popularity of participants, citing factor, initiator-media link	✓

The difference of all the aforementioned algorithms is summarized in Table 2.2. The aforementioned approaches did not take into account the object node which is influenced. In contrast, both of our work *blog cascade affinity* (Chapter 3) and CASINO (Chapter 4) investigate the object node. Moreover, only OOLAM takes into account the features that are driven by social influence effect. In detail, they evaluated the influence of nodes. In contrast, in CASINO we not only study the influence of u on the sign of edge $\overrightarrow{uv}$ but also investigate the conformity of v and its effect on u's influence; in *blog cascade affinity*, we investigated a series of social influence driven features such as *number of friends*, *popularity of participants*, *citing factor*, and *initiator-media link*.

2.1.3 Object Node

The majority of existing work focused on either the subject who is influencing others or the edge where influence propagate. They have not systematically investigated the object in social influence phenomenon. The only work regarding to the object node who is influenced by others can be found in viral marketing research. The majority of marketers are interested in evaluating the probability for a user to purchase a particular product. It is of much importance for on-line

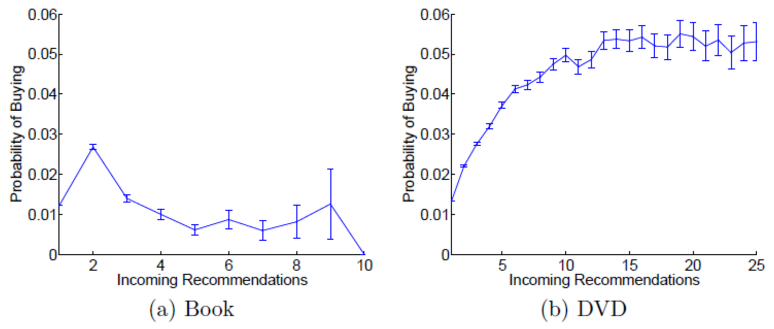


Figure 2.2: Affinity of buying a product as a function of number of received recommendations.

advertising and information propagation. Authors in [10] have investigated people's buying behavior towards the number of recommendations. In Figure 2.2 the authors showed that the probability for a user to buy a product versus the number of recommendations she had received. For the case of book, the buying probability exhibits a decrease as the number of recommendations increases while the buying behavior for DVD keeps on increasing until a saturation point after 10 recommendations. However, apart from the number of recommendations, many other features that are related to social influence study need to be taken into account in this field of research. In Chapter 3 we shall propose a series of novel features that exert significant impact on the probability for a blogger to join a cascade.

In summary, all the aforementioned work in the field of individual-level study have only focused on the subject node and social tie in social influence phenomenon. They have ignored the object node. In this thesis, we address two specific problems with respect to the object nodes. Firstly, we propose to study and predict the probability for a blogger to join a cascade. Secondly, we propose a novel algorithm CASINO to evaluate the influence and conformity of nodes in a network in order to measure the nodes' inclination to be influenced by others.

2.2 Community-level Analysis

As an effect of social influence phenomenon, people form *communities* in a network. Consequently, a great deal of work have focused on mining implicit communities in online social networks [57]. A *community* is a group of people with some common properties. Community mining is in fact subgraph identification [58] or node clustering [59]. There are two main research directions in this area: community evolution study and community affinity.

2.2.1 Community Evolution

The basic problem to be addressed with respect to community-level study is how to evaluate and extract communities which are highly evolving. To solve this problem, researchers made a well accepted assumption that communities are groups of people who are relatively stable along the evolution of network [32]. Based on this assumption, many frameworks have been proposed to find communities within evolutionary networks [38]. We present some of them in the following.

FacetNet. Lin et al. [33] analyzed communities and the evolution of them through a unified process. An *adjacency matrix factorization* approach is introduced in the paper. According to the approach, the adjacency matrix of a network W can be factorized as $W = X\Lambda X^T$ where $X \in R_+^{n\times m}$ and $\sum_i x_{ij} = 1$ [60]. In addition, Λ is an $m \times m$ non-negative diagonal matrix. According to the paper, matrices $X\Lambda$ fully characterize the community structure in the network. Based on the factorization, the authors proposed *snapshot cost* which captures how well the community structure $X\Lambda X^T$ fits W at time t. It can be expressed as $\mathcal{CS} = D(W\|X\Lambda X^T)$ where $D(A\|B)$ is the KL-divergence between A and B. Similarly, the authors proposed another concept called *temporal cost* to measure how consistent the community structure at time t is with respect to that of $t-1$. It can be calculated as $\mathcal{CT} = D(Y\|X\Lambda)$ where $Y = X_{t-1}\Lambda_{t-1}$. Both of the cost are then linearly combined together into a total cost formula as follows.

$$cost = \alpha \cdot D(W\|X\Lambda X^T) + (1-\alpha)\cdot D(Y\|X\Lambda) \qquad \text{(Eq. 2.15)}$$

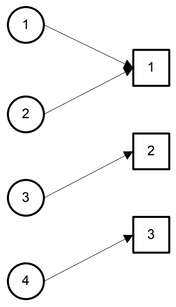
Following this approach, communities can be detected by minimizing this *cost*. The authors justified this model in both synthetic dataset and real-world datasets.

GraphScope. GraphScope [37] is a parameter-free algorithm where the Minimum Description Length (MDL) principle is employed to extract communities as well as their changes. According to this model, the evolution of a graph G is modeled as a graph stream $\mathcal{G} = \{G^{(1)}, G^{(2)}, \ldots, G^{(t)}, \ldots\}$. The goal of the model is to find good partitions of source and destination nodes that describe the community structure. In order to evaluate the quality of partitions, they proposed an encoding scheme to represent temporal partitions, graphs and subgraphs. For example, Figure 2.3(a) depicts a social network where circles represent source node and square denotes destination nodes. The adjacency matrix of the graph is shown in Figure 2.3(b). Conceptually, such a binary matrix can be store as a binary string with length mn, along with the two integers m and n. For example, Figure 2.3(b) can be stored as 110000100011 (in column major order), along with two integers 4 and 3. To reduce the space, more scientific scheme such as Huffman coding can be employed to store that string. The code length for that is accurately estimated as $mnH(G^{(t)})$ where $H(G^{(t)})$ is the entropy of binary string of $G^{(t)}$ that can be computed as follows.

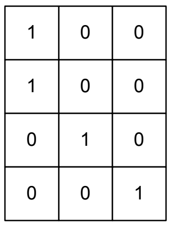
$$H(G^{(t)}) = -p(1)\log p(1) - p(0)\log p(0) \qquad \text{(Eq. 2.16)}$$

In the equation, $p(1)$ (*resp.,* $p(0)$) denotes the ratio of 1 (*resp.,* 0) in the entrance of matrix $G^{(t)}$. After encoding the subgraphs using this method, MDL is employed to evaluate the cost between the code of different subgraphs. In this way, those subgraphs with the minimum cost can be view as communities as they do not change much between snapshots.

MONIC. MONIC [28] models the changes within each individual community. The authors first clustered the graphs at each snapshot. The goal of the model is to find similar clusters across different snapshots of the same graph. These similar clusters are viewed as the same community evolving from one snapshot to another. In order to do this, they proposed a measure



2.3.a:



1	0	0
1	0	0
0	1	0
0	0	1

2.3.b:

Figure 2.3: Coding of GraphScope.

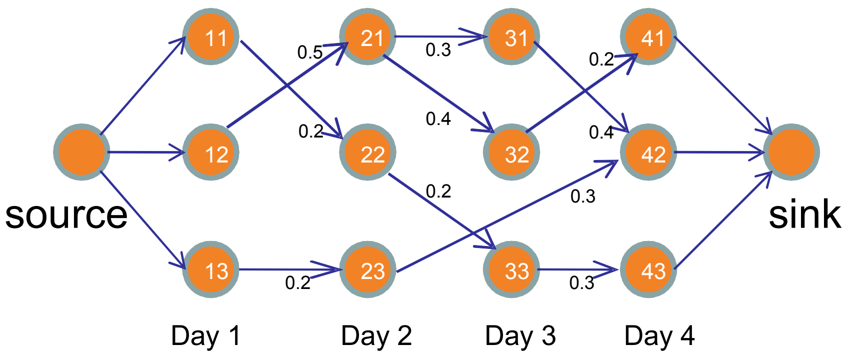


Figure 2.4: Cluster evolution graph.

to evaluate the overlap from a cluster X at time t_i to cluster Y at time t_j ($t_i < t_j$) which can be described as the following.

$$overlap(X,Y) = \frac{\sum_{a \in X \cap Y} age(a,t_j)}{\sum_{x \in X} age(x,t_j)} \qquad \text{(Eq. 2.17)}$$

The $age(x,t_j)$ describes the weight of record x at time t_j. Using this measure the best matching cluster, which exhibits the highest overlap value with X, can be selected. The selected cluster is then viewed as the identity cluster that evolves from X. The authors also identified a set of key events, such as *survive*, *split*, *disappear*, which are further studied based on the changes of clusters.

Stable cluster. Bansal et al. [30] also viewed the evolution of a graph as a stream of graphs and clustered each snapshot into partitions. They further used jaccard similarity to measure

Table 2.3: Summary of community dynamics research.

Models	*Difference measure*	*Parameter-free*	*Evolution pattern comparison*
GraphScope [37]	minimum description length	✓	X
MONIC [28]	cluster intersection	X	X
Stable Cluster [30]	jaccard similarity	X	X
FacetNet [33]	KL-divergence	X	X
Event-driven [61, 65]	overlap ratio of vertices	X	X
AffRank	ΔAffinity and ΔAffinity rank	✓	DTW distance

the similarity between clusters at different snapshots. It is computed as the intersection of community members at different time points divided by the union of the members.

$$overlap(C_1,C_2) = \frac{|C_1 \cap C_2|}{|C_1 \cup C_2|} \tag{Eq. 2.18}$$

In this way, a *cluster evolution* graph is formed. An example is shown in Figure 2.4. Each node is a cluster at a particular time point while edge weight is the jaccard similarity between clusters. The nodes in column "Day 1" represent the clusters identified in the graph at first snapshot. An edge pointing from node "12" to node "21" indicates that the cluster C_{12} is similar to C_{21} with similarity of 0.5. Such a high similarity means that C_{21} is probably evolved from C_{12}. Thus, from this graph it is easy to find the most stable cluster by tracing each path from "source" to "sink" and select the one which exhibits the highest accumulated weight.

Event-driven. Asur et al. [61, 65] studied the evolution of graphs to understand particular patterns for communities and individuals over time. They defined the overlap ratio of vertex set for the same community over two timestamps as $overlap(x,y) = \frac{|x \bigcap y|}{\min(|x|,|y|)}$. *Popularity index* for community j at time i is:

$$PI(C_i^j) = \sum_{x=1}^{V_i} Join(x, C_i^j) - \sum_{x=1}^{V_i} Leave(x, C_i^j) \tag{Eq. 2.19}$$

Influence index for node x is defined as: $Inf(x) = |moves(companions(x))| / |moves(x)|$.

The aforementioned work have proposed different ways to evaluate the evolution of a community. They are summarized in Table 2.3. Existing work all focused on mining the similarity of communities from the interaction of their members. In contrast, we propose a series of novel measures in *AffRank* (Chapter 5) by evaluating the similarity between the evolution pattern of communities. We show by experiments that these measures exhibit significant effect in predicting the *affinity rank* of communities.

2.2.2 Community Affinity

Another area of community-level study is the finding the factors that exert effect on a community's ability to attract new members. Formally, the ability of a community to attract new members is referred to as *community affinity* [45, 62]. We describe some representative work which investigate *community affinity*.

Group formation. Backstrom et al. [38] demonstrated that the community size, connectivity between community members, and number of friends a user has in a community have strong influence on *community affinity*. They modeled the affinity problem as a standard classification task: the nodes eventually join a group are denoted as positive while those do not are denoted as negative samples. Two real-world social network datasets (*LiveJournal* and *DBLP*) were studied in that paper. They extracted several features (friends in the group, clustering coefficient of the group, etc.) for each node and employed a decision tree to predict the sign of node samples. As shown in Figure 2.5, having a large clustering coefficient is negatively related to growth.

The experimental results showed that *community affinity* is highly affected by existence of friends in the target community.

Dynamics in VM. Leskovec et al. [63] showed that an individual's probability of buying a DVD increases with the number of recommendations he has received. There is a *saturation point* at the value of 10, which means after a person receives 10 recommendations on buying a

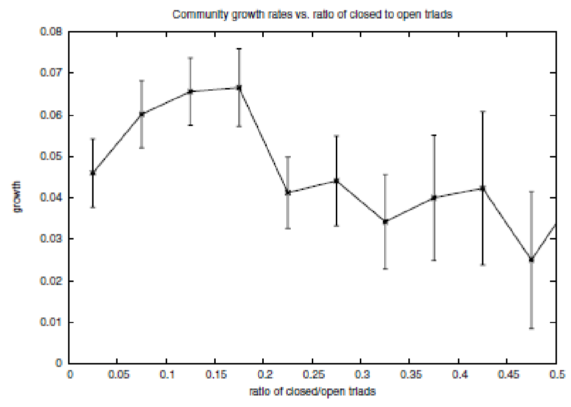


Figure 2.5: The ratio of community growth as a function of clustering coefficient

particular DVD, the probability of buying does not increase anymore. Moreover, by studying the correlation between people's buying behavior and the number of recommendations they have received the authors found that such correlation is significant in some products (*i.e.,* DVD and Book). Further, a logistic regression model is employed to test the success of recommendation based on the findings on the affinity of buying a product.

Propagation in Flickr. Cha et al. [41] conducted a study on *Flickr* over the same problem. The authors investigated the correlation between picture's popularity and network topology. Specifically, the number of fans for each picture at different distance from the uploader is examined in the paper. It is reported that the probability for a user to become a fan of a photo increases with the number of her friends who are already fans of the photo. On the other hand, the authors also reported that the number of fans a picture have is also affected by the elapsed time since the picture was uploaded.

Table 2.4 summarizes the differences between *AffRank* we propose in Chapter 5 and existing approaches. Obviously, there exists a correlation between the *number of recommendations*, *clustering coefficient* in a community and the community's ability to absorb new members. These work have showed that the *number of recommendations* feature exerts positive effect on community affinity while *clustering coefficient* exerts negative effect. The aforementioned

Table 2.4: Summary of community affinity research.

Models	***Features adopted***	***Method***	***Target network***	***Affinity evolution study***
Group formation [38]	friends in the group, group size, clustering coefficient	decision tree	collaboration network	X
Dynamics in VM [63]	#recommendations, price, #reviews	logistic regression	recommendation network	X
Propagation in Flickr [41]	#fans, #friends, elapsed time	statistic analysis	flickr network	X
AffRank	affinity rank history, affinity evolution distance, average rating besides above features	ARX	social rating network	✓

work did not address the issue of predicting the future affinity ranks of products in a product community which is more valuable in many applications. Are there any other features that will affect a community's ability to absorb new members? In this thesis, we propose a novel model in Chapter 5 taking into account the evolution of community affinity and the difference of affinity evolution patterns between communities in order to rank communities according to their ability to attract new users in the near future.

2.3 Network-level Analysis

Social influence phenomenon has turned social networks into important channels for information propagation. The dynamics of information dissemination in social networks is of paramount importance in processes such as rumors or fads propagation, spread of product innovations or "word-of-mouth" communications. Due to the difficulty in tracking a specific information when it is transmitted by people, most current understandings of information spreading in social networks come from models of indirect measurements. It is strongly related to the research of epidemic and contagion problems which study the propagation models of viruses and diseases.

A main research goal in this field is the *influence maximization* problem. Formally, the problem can be described as follows, find the set of initial users of size k (referred to as *seeds*) so that they eventually influence the largest number of individuals (referred to as *influence spread*) in the network. The *influence spread* in the network may follow one of the following cascade models.

- *Independent cascade (*IC*) model.* Let A_i be the set of nodes that are influenced in the i-th round and $A_o = |S|$. For any $(u,v) \in E$ such that u is already in A_i and v is not yet influenced, v is influenced by u in the next $(i+1)$-th round with an independent probability p, which is referred to as the *propagation probability*. Thus, if there are t neighbors of v that are in A_i, then $v \in A_{i+1}$ with probability $1-(1-p)^t$. This process is repeated until A_{i+1} is empty.

- *Weighted cascade (*WC*) model.* Let $(u,v) \in E$. In this model, if u is influenced in round i, then v is influenced by u in round $(i+1)$ with probability $1/v.degree$. Thus, if v has t neighbors influenced at the i-th round then the probability for a node v to be influenced in the next round is $1-(1-1/v.degree)^t$.

- *Linear threshold (*LT*) model.* In this model, each node v has a threshold θ_v uniformly and randomly chosen from 0 to 1; this represents the weighted fraction of v's neighbors that must become influenced (active) in order for v to be influenced. All nodes that were influenced in step $(i-1)$ remains so in step i, and any node v is influenced when the total weight of its influenced neighbors is at least θ_v.

To achieve the goal of *influence maximization*, existing work have proposed many methods to find some nodes in the network to spread the information initially with minimal cost and maximal influence. The algorithmic study towards the problem of influence maximization within social networks can be traced back to the year 2001 when Domingos et al. [64, 66]

proposed a probabilistic method to predict the number of influenced nodes in a network to help companies determine the potential customers for marketing a product. They modeled the customers as a social network and adopted *markov random field* method to study the propagation of influence. After that, many algorithms have been proposed to solve the problem. These algorithm can be classified into two groups, greedy algorithms and heuristic algorithms.

2.3.1 Greedy Algorithms

Greedy algorithms are a group of greedy approaches which greedily select the node with the maximum marginal gain towards the existing seeds in each iteration. These algorithms provide high quality results which are within 63% of the optimum solution. However, they suffer from the running time. The state-of-the-art technique [42] requires more than 14 hours to find seed set of size 100 in a network with only 37,154 nodes.

General greedy. Consider an arbitrary function $f(\cdot)$ that maps subsets of a finite ground set U to non-negative real numbers. Then f is *submodular* if it satisfies a natural "diminishing returns" property: the marginal gain from adding an element to a set S is at least as high as the marginal gain from adding the same element to a superset of S. Submodular function have been studied extensively in the field of mathematics [67]. Kempe et al. proved that several popular cascade models are submodular (*i.e., Independent cascade model*, *Weighted cascade model* etc.). Particularly, if the cascade model is fixed and the influence function is proved to be submodular and monotone, then they showed that the influence maximization problem can be solved by starting with an empty set and iteratively selecting the nodes which achieve the best marginal gain towards the current seeds. Note that although the minimization of a submodular function is shown to be within polynomial time [68, 69], the maximization of that is proved to be NP-hard. Hence, the influence maximization problem is also NP-hard.

Several work has focused on designing approximated algorithms to achieve the maximization task in polynomial time [70]. Kempe et al. [4] also proposed an approximate algorithm based on greedy strategy. According to [71], if a greedy maximization algorithm

Table 2.5: Time complexity of different algorithms.

Algorithm	*Time complexity*
General Greedy	$O(knRm)$
CELF	$O(knRm)$
MixGreedy-IC	$O(kRm)$
MixGreedy-WC	$O(kTRm)$
DegreeDiscount-IC	$O(klogn+m)$
MSA	$O(Tk\bar{d})$
PMIA	$O(nt_{i\theta}+kn_{o\theta}n_{i\theta}(n_{i\theta}+\log n))$
MIA-N	$O(nt_o+kn_on_i(\min(k,h_{max})n_i+\log n))$
CGA	$O(m+nRlm'+klRm'+kRm')$
IMCD	$O(\|A\|nm^2+k\|A\|m^2)$
PATINA-IC	$O(kRm')$
PATINA-WC	$O(kTRm')$
CINEMA	$O(k'm'n'+kTRm')$

of a submodular function f returns the result A_{greedy}, then the following holds $f(A_{greedy}) \geq (1-1/e)\max_{|A|\leq k} f(A)$. That is, a greedy algorithm can give near optimal solution to the problem of maximization of a submodular function. Accordingly, Kempe et al. guaranteed that their greedy algorithm can achieve influence spread within $(1-1/e)$ of the optimal influence spread. However, the proposed algorithm takes $O(knmR)$ time to solve the influence maximization problem (Table 2.5), which is computationally very expensive for real-world social networks.

CELF (Cost-Effective Lazy Forward). In order to reduce the computation time of influence maximization problem, Leskovec et al. [43] proposed an algorithm called CELF (Cost-Effective Lazy Forward) that is reported to be 700 times faster than the simple hill-climbing algorithm proposed by Kempe et al. on real networks. The CELF is also based on the submodular property of the cascade influence function. They observed that in each round, the hill-climbing algorithm needs to recompute the influence $\sigma(v)$ of each node v. In most cases, the *marginal gain* of a node v, given by $\sigma(v|S) = \sigma(S \cup \{v\}) - \sigma(S)$, may not change significantly between consecutive rounds. So instead of recomputing the spread for each node at every round of seed selection, CELF performs a *lazy* evaluation. Initially, it marks all $\sigma(v|S)$ as *invalid*. When selecting the next seed, it scans the nodes in decreasing order of their $\sigma(v|S)$. If the $\sigma(v|S)$ for the top node is *invalid*, then CELF recomputes it and inserts it into the existing order of the $\sigma(v|S)$ (e.g., using a priority queue). In many cases, the recomputation of $\sigma(v|S)$ will lead to a new value which is not significantly smaller. Consequently, often the top element will remain at the top even after recomputation. In the worst case, during each selection CELF needs to recompute the marginal gain for all the remaining nodes resulting in a worst-case time complexity of $O(kmRn)$ (Table 2.5).

MixGreedy. Chen et al. [42] reduced the computation of marginal gain from $O(mn)$ to $O(m)$. To compute the influence of each node, they process the graph by removing unnecessary edges according to the cascade model such that computing the marginal gain for all the nodes only requires a linear traversal over the network. Their *random removal* method can be summarized as following. Since in *independent cascade (*IC*) model* each edge has the probability p to take effect in the cascade, they randomly remove each edge in the graph G with probability $1-p$. In this way, G is separated into pieces and each piece is the scope of the node v's influence spread within it. Thus, computing the marginal gain of a node will only require a linear traversal of the scope. As a result, computation of the marginal gain of a node only requires $O(m)$ operations which is the complexity of removing edges from G. Similarly, when the network follows *weighted cascade (*WC*) model*, they remove each edge with probability $1 - 1/v.degree$. The influence of each node can be computed by adding the gain in R iterations of random removal process. Based on this they proposed the *MixGreedy* algorithm which follows the random removal process in computing the marginal gains and then utilizes the CELF approach for updates. The time complexities of the *MixGreedy* approach for the aforementioned two cascade models are $O(kRm)$ and $O(kTRm)$, respectively (Table 2.5). They empirically demonstrated that the running time of *MixGreedy* is smaller than CELF.

CGA (Community-based Greedy Algorithm). Wang et al. [72] proposed a community-based greedy solution to the problem. In order to reduce the running time, they first detect communities based on IC model and then mine the top-K nodes across communities. They developed a cost function that optimized the community assignment in mobile networks. Our work in Chapter 6 differentiates from this one in the following points. Firstly, instead of designing a IC model-aware community detection method, we adopt existing network partition models which not only can be applied to all cascade models but also exhibit smaller time complexity. In fact, their community detection process under IC model takes $O(m+nRlm'+klRm')$ which is time consuming in huge network, while the community detection module in our work only takes only $O(m)$ time. Secondly, CGA algorithm selects each influential node based on a unified optimization formula, which requires the information for all communities stored in a unified space. Thus, it is not easily applied in distributed environment. In contrast, our work in Chapter 6 has distributed the node selection task into the sub-networks and is easily applied in distributed environment. Empirical study has also shown that our work can speed up the algorithm by about 3 times if applied in a 5-slaves distributed environment.

IMCD (Influence Maximization under Credit Distribution model). Goyal et al. [73] proposed a *credit distribution (*CD*) model* that leverages on historical *action logs* of a network to learn how influence flows in the network and use this to estimate influence spread. An *action log* is a set of triples (u,a,t) $\in A$ which says user u performed action a at time t. The basic idea is that if user v takes action a and later on v's friend u does the same, then the authors assume that action a have propagated from v to u. Based on this assumption the CD model assigns "credits" to the possible influencers of a node u whenever u performs an action. The sophisticated variant of this model distinguishes between different influenceability of different users by incorporating a *user influenceability function*. It is defined as the fraction of actions that u performs under the influence of at least one of its neighbors (*e.g., v*) and is learnt from the historical log data. In contrast to our approach, this model suffers from two key limitations. Firstly, it depends on the availability of large amount of historical action logs to compute influence probability as well as user influenceability. Unfortunately, historical action logs may not be available to end-users in many real-world social networks.

2.3.2 Heuristic Algorithms

As the greedy algorithms are time consuming, a series of heuristic algorithms have been proposed recently which save much time. Instead of computing the marginal gain of nodes in each iteration, these heuristic algorithms iteratively select nodes based on a specific heuristic, such as *degree*, *PageRank* [4]. However, the result generated in these approaches are not satisfactory. Researchers have investigated several other heuristics that may improve the result quality of the algorithm.

DegreeDiscountIC. The running times of the aforementioned greedy approaches are still large and may not be suitable for very large social networks. Hence, Chen et al. [42] used *degree discount* heuristic, where each neighbor of newly selected seed discounts its degree by one, to reduce the running time. Although this heuristic can be used for all cascade models, they enhanced the degree discount heuristic to make it suitable for the independent cascade model. They demonstrated that the heuristic-based approach is orders of magnitude faster than all greedy algorithms. However, the seed set quality can be inferior compared to the greedy approaches. Recently, they proposed a new heuristic approach [44] which introduced a parameter to control the balance between the result quality and running time. However, the result quality is much lower than greedy approach although it has improved much over *degree discount*. It is worth mentioning that although the importance of reduction in computation time is undeniable, the seed set quality is more significant to companies as ultimately they would like to maximize the influence spreads of their new products in order to reach out to largest possible customers base.

MSA and SASH (Simulated Annealing Based Influence Maximization). Jiang et al. [74] proposed an algorithm based on *Simulated Annealing* (SA) for the influence maximization problem. It is the first SA-based algorithm for the problem. Additionally, two heuristic methods are proposed to accelerate the convergence process of the algorithm. The algorithm is developed specifically for IC model, it initiates the seeds set by randomly selecting k nodes. In each iteration afterwards, a node in the current seed set is replaced by another one which are not in the seeds, thus a new seed set is formed. If the new seed set can generate better influence spread than the old one under IC model, the seed set is updated to the new one. This process is iterated for T times until converge. The time complexity of MSA is $O(Tk\overline{d})$ where $\overline{d}$ denotes the average degree of nodes. Experimental results have shown that the two heuristic methods MSA and SASH generates better result quality comparing with *degree discount* algorithm. The running time of MSA and SASH is similar with that of *degree discount*. However, the improvement in result quality is limited (*i.e.,* 3% to 8%). Comparing with this, our PATINA algorithm generates much better result than *degree discount* (*i.e.,* 60%). On the other hand, MSA and SASH are restricted to IC model, while PATINA can be applied to both IC and WC model. Moreover, MSA and SASH algorithms[1] have not taken into account that individuals may exhibit different influence on different topics, which is addressed in our CINEMA algorithm.

PMIA, LDAG and SimPath. Chen et al. proposed PMIA technique [44] over IC model, which selects a limited number of paths that satisfy a given threshold θ to compute the influence. This threshold is used to tune the tradeoff between influence spread and running time. The authors compared CELF, PMIA and degree discount-based approaches and demonstrated that PMIA improves the influence spread generated by *degree discount* by 3.9%-6.6% over *Hep* dataset. However, the running time of PMIA is an order of magnitude slower than the degree discount-based technique with time complexity of $O(nt_{i\theta} + kn_{o\theta}n_{i\theta}(n_{i\theta} + \log n))$ where $t_{i\theta}, n_{i\theta}, n_{o\theta}$ are constants decided by θ. Based on that, another model called LDAG [75] is developed. It is similar to PMIA except that LDAG is specifically designed for the *linear threshold* model. LDAG also performs slightly better than *degree discount* over the *Hep* dataset in terms of influence spread. More importantly, the authors demonstrated that CELF still outperforms all these three heuristic-based approaches with respect to quality of influence spread. In order to improve LDAG further, another alternative model called *SimPath* is developed [76]. Similar to

[1]Note that despite our best efforts (including contacting the authors), we could not get the source code of MSA and SASH

LDAG, it is also designed for selecting influential seeds under *linear threshold*. However, unlike LDAG, *SimPath* computes the spread by exploring simple paths in the neighborhood. Using a parameter η, the tradeoff between running time and seeds quality can be controlled. It have been shown by experimental results that by selecting proper parameter η, *SimPath* outperforms LDAG both in efficiency and accuracy. Both LDAG and *SimPath* are algorithms designed specifically for *linear threshold* model. PATINA (*resp.,* CINEMA) is designed for IC and WC (*resp.,* C^2) models. On the other hand, both LDAG and *SimPath* are heuristic-based algorithms that do not guarantee the result quality. However, PATINA and CINEMA are greedy-based algorithms that generates result with good quality that is within 63% of the optimal.

MIA-N. The aforementioned approaches have investigated different ways to maximize the influence under IC, WC and LT models. However, all of the existing models ignore an important aspect of influence propagation. Not only positive opinions on products and services that we receive may propagate through the network, negative opinions are also propagating. To this end, Chen et al. [77] proposed a novel model called IC-N (Independent Cascade Model with Negative Opinions) which introduces a *quality factor* to control the negative opinion propagation probability. In order to maximize the influence under IC-N model, a new heuristic algorithm called MIA-N, which borrows the core idea of PMIA, is developed. It defines a *maximum influence in-arborescence* to estimate the influence to an arbitrary node v from other nodes. The time complexity of MIA-N algorithm is $O(nt_o + kn_on_i(\min(k, h_{max})n_i + \log n))$ where h_{max} denotes the maximum height of the *maximum influence in-arborescence*. Experimental results have shown that the heuristic algorithm generates influence spread very close to greedy method while is orders of magnitude faster than greedy approach. Although IC-N incorporates negative opinions in networks, it suffers from another limitation. It assumes each node have the same influence and consequently exhibits the same *quality factor* that negative opinions can emerge. However, in real social networks individuals may exhibit different probabilities to express opposite opinions. Moreover, individuals may show different influence in different topics. To this end, we propose in Chapter 7 a novel *Conformity-aware Cascade model* (C^2) which takes into account the *influence* and *conformity* of nodes while incorporating negative opinions propagation in networks. In fact, the *quality factor* proposed in IC-N model to control the negative opinion propagation probability can be viewed as a special case of conformity where an individual negatively following another. However, *quality factor* does not completely capture the negative opinions propagation in that it assumes individuals show the same *quality factor*. C^2 model addressed this problem by computing a pair of influence and conformity indices for each individual.

Unlike greedy algorithms, the quality of influence spread of these models are not guaranteed to be within 63% of the optimal. Although the aforementioned heuristic algorithms save much time than greedy algorithms and improved the result quality compared to *degree* and *PagaRank* heuristics, the results generated from these heuristic algorithms have been showed poorer than the greedy algorithms.

Existing work in influence maximization suffers either from the running time (*i.e.,* greedy approaches [4, 42, 43]), or the result quality (*i.e.,* heuristic approaches [42, 44]). In order to address both these limitations, we propose a novel algorithm PATINA in Chapter 6 by solving the problem in a parallel and distributed way. Thus, we can complete the maximization task in a shorter time while preserving good quality similar to the previous greedy approaches [4, 42, 43].

2.4 Summary

Compared with the existing work, our research has the following novelty.

- In the field of individual-level study, our work on *blog cascade affinity* in Chapter 3 investigates the probability for a blogger to join a cascade. Traditional work in the field of individual-level social influence study all aim to find the nodes having the most influence on others. Instead, we found the individuals which are most probable to be influenced by others using a group of network features and social characteristics. Besides, we are studying social behavior using both local network topology and social characteristics. We also take into account both the evolution of communities a person participated in and the global network trend which we believe to have correlation with individual behavior. Besides, in Chapter 4 we also investigated the influential mining problem by taking into account both factors in social influence phenomenon, namely, influence and conformity. To the best of our knowledge, none of the existing work in this field have systematically considered both factors together.

- In the field of community-level study, existing work have investigated ways to detect and evaluate the change of communities. However, how to evaluate and rank the communities' ability of growing is not answered yet. In this thesis, we study the evolution of product communities and propose *AffRank* in Chapter 5 to rank products according to their ability to absorb new users.

- In the field of network-level study, existing work have found different evolution phenomenons in networks and tried to utilize these phenomenon in information propagation. One hot topic in this field is *influence maximization* problem. Many algorithms have been proposed to address this problem. However, existing greedy algorithms are time consuming in large networks. In this thesis, we propose a novel algorithm in Chapter 6 by partitioning the whole network into sub-networks where social influence can hardly propagate between each other. In this way, the problem can be solved in parallel and distributed platform. Hence, it can reduce the running time while preserving almost the same result quality as existing approaches. On the other hand, existing algorithms assume influence propagation follows IC, WC or LT models. These models have not taken into account the influence index and conformity index of nodes which exert significant impact on the influence propagation between a pair of nodes. In this thesis, we propose in Chapter 7 a novel cascade model (C^2) which incorporate both influence and conformity indices in cascade model. To solve the *influence maximization* problem under C^2 model, we propose a conformity-aware PATINA algorithm, namely, CINEMA. In contrast with the work in [73], CINEMA does not require any historical action logs. Secondly, similar to existing work, the CD model does not incorporate conformity and context information in computing influenceability of a node.

Chapter 3

Blog Cascade Affinity, Analysis and Prediction

In last chapter, we reviewed representative work in all the three levels of social influence study and discussed their limitations. In this chapter, we present our work on blog cascade affinity with respect to individual-level social influence study.

The popularity of blogs has been increasing dramatically over the last few years. According to a recent report by Technorati [1] [78], a popular blog search engine, more than a half of the Internet users read blogs. Technorati have indexed more than 133 million blogs since 2002, and have tracked blogs in 81 languages by June, 2008. Blogs contain diverse variety of information. General topics include personal diaries, experiences, opinions, information technology, and politics to name a few. Due to their accessible and timely nature, many bloggers surveyed have advertisement on their blogs. The mean annual revenue for blogs with advertisement is estimated to be \$6,000 [78]. This figure jumps to \$75,000 when we consider only those blogs having 100,000 or more unique visitors per month.

The rest of this chapter is organized as follows. Firstly, we present the motivation of this work in Section 3.1. In Section 3.2, we introduce the dataset as well as the cascade extraction process. The macroscopic and microscopic cascade features and their analysis are described in Sections 3.3 and 3.4, respectively. Section 3.5 describes our proposed models to measure

[1] http://technorati.com

and rank the probability of a blog to join a cascade. In Section 3.6, we conduct an exhaustive empirical study to evaluate many aspects of our proposed techniques and their effectiveness. The last section concludes the chapter. A preliminary version of this chapter has been published in [79].

3.1 Motivation

A blog consists of several entries. Each entry within a blog, called a *post*, is time stamped and the most recent entries always appear at the top. Bloggers can also create hyperlinks to other blogs or websites in their posts. The universe of all these blogs and their interconnections is often referred to as *blogosphere* [80, 78]. Blogosphere is an intuitive source for data involving the spread of information and influence within the network of bloggers [19, 20, 32, 60, 80, 81, 82, 83]. By analyzing the linking patterns from one blog post to another, we can infer the way information is propagated through the blog network over the Web. In particular, a piece of information flows from a post to another along the hyperlink between them. For example, consider Figure 3.1(a). The ellipses represent different blogs (*e.g.,* b_1, b_2, b_3, b_4, b_5, and b_6), and each ellipse contains a set of posts. The edges in the figure indicate hyperlinks between posts. Assume that post p_1 in blog b_1 contains opinion about recent events related to the spread of H1N1 *virus*. Some time later, blog b_2 visited b_1 and wrote a post p_2 in response to this topic of discussion and explicitly created a hyperlink to p_1. Subsequently, new posts will join this conversation by linking to existing posts. For instance, at time T_0, the structure of this conversation related to H1N1 virus containing a group of posts (p_1, p_2, p_3, and p_4) is depicted by the dashed rectangular component in Figure 3.1(a). Aggregating all the linked posts by backtracking the hyperlinks will result in a DAG (Directed Acyclic Graph), where each node is a post. Such a DAG is called a *cascade* [1, 84] (also known as *conversation tree*). Cascade is the most common phenomenon of information propagation within blogosphere. All posts in the *same* cascade typically discuss about a similar topic.

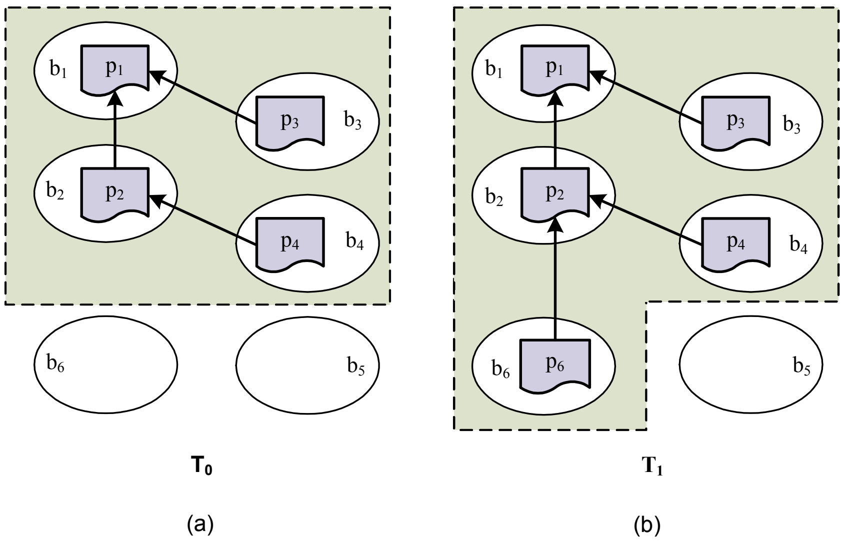


Figure 3.1: Blog cascade.

Observe that at time T_0 there are two blogs, b_5 and b_6, which did not join the conversation on H1N1 virus by writing a post and linking to the cascade. Now assume that at time $T_1 > T_0$ b_6 joined the cascade by writing a post p_6 and linking it explicitly to p_2. The modified structure of the cascade is now depicted in Figure 3.1(b). Notice that b_5 still did not join the conversation. Why did b_6 join the cascade but b_5 did not? Is it possible to predict the *cascade affinity* of b_5 and b_6 by analyzing the information embedded in the cascade at time T_0? In order to provide answers to these questions, *in this chapter we propose probabilistic and non-probabilistic techniques to analyze an array of macroscopic and microscopic cascade features for predicting which blogs are highly likely to join the cascade in the future*. We refer to the phenomenon of a blog's inclination to join a specific cascade as *cascade affinity*.

Although the notion of information cascade was formally introduced by Sushil Bikhchandani [85], it was first systematically studied in the context of blogosphere by Kumar et al. [32]. Majority of research on blog cascades [1] have focused their attention at the *macroscopic* level. In particular, these efforts investigated information flow in cascades, common shapes of cascades and their frequencies, and performed a series of topological analysis. In contrast, we take a hybrid view by analyzing cascade affinity behavior of individual bloggers from both *microscopic* and *macroscopic* level [79]. As blog cascading behavior happens at individual level when a blogger decide to reference another's post, it is a personal behavior for each blogger. Studying individuals' personal behavior without taking into account their personal preferences or local social relationships may not be accurate. To this end, besides *macroscopic* ones we propose to use a series of *microscopic* feature to study blog cascade. According to the experimental result in Section 3.6, taking into account a *microscopic* feature, namely *number of friends*, can improve the prediction performance by around 9 times. *To the best of our knowledge, our work is the first approach that undertakes a systematic study to predict such behavior.*

3.1.1 Applications

The knowledge of a blogger's affinity to cascades is useful in several applications. It not only facilitates the design of advanced blogging system with more sophisticated personalized recommendations and filters, but also help us to set up intelligent strategies in online advertising. By predicting which blogs have stronger affinity to a cascade, we can make recommendations to those bloggers in case they have not yet read any post in the cascade. Consequently, we can influence the population faster by accelerating the information propagation process. In this way, new services or products can be disseminated and popularized in a shorter time. Furthermore, we can predict to what scale of population a cascade will finally expand so that when disseminating an advertisement along blog cascade we can understand the final effect of the advertisement ahead of time and adjust our advertisement strategies accordingly. For example, assume that the release of "iPhone4" may trigger off a cascade within the blogosphere. At the very beginning, there is no post talking about *iPhone4* at time t_0 (see Figure 3.2). Later at time t_1 there are two posts talking about this topic which initiates a cascade over "iPhone4". By studying the cascade affinity of many candidate blogs who are most probable to join this cascade, we can predict that two new bloggers have high chance to join this cascade at time

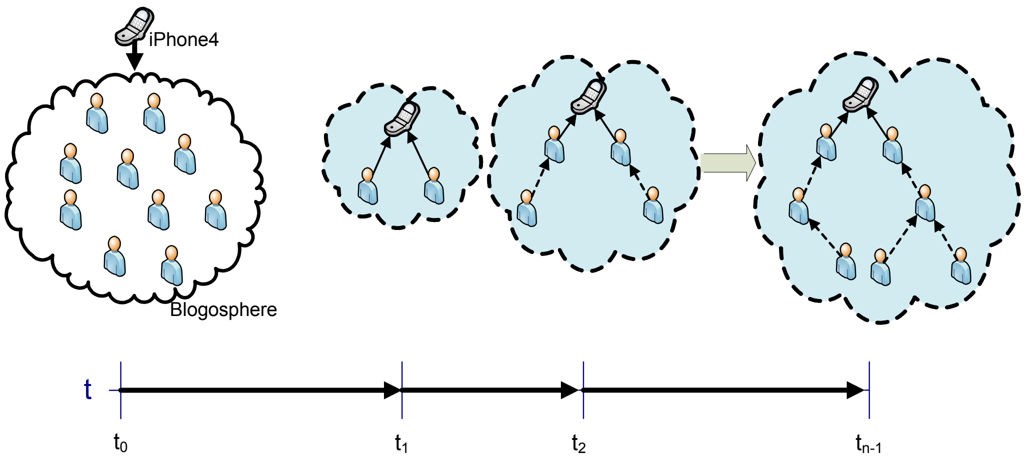


Figure 3.2: An example application of blog cascade affinity prediction.

t_2. In Figure 3.2, a person with a dashed outgoing arrow represents a blogger who is predicted to join the cascade at current timestamp whereas a solid arrow denotes the blogger who has already joined it. Thus, we may estimate the final scope of the cascade at t_{n-1} by iteratively predicting the set of bloggers who may join it at the next timestamp. Our ability to forecast the final scope of the population involved in this cascade at an early stage paves way to more judicious adjustment of advertisement strategies and budget ahead of time. For instance, without loss of generality, we assume there is a budget for online-advertisement. With the help of the knowledge from blog cascade affinity study, we can benefit from the following two aspects. Firstly, we can easily know which blogger may trigger a large cascade that makes the advertisement attached to it reach a large population. Thus, we can maximize the expected income from the limited budget by selecting particular sites to put the advertisement. Secondly, as long as we know the final scope of population that the advertisement can reach, we can easily know the maximum benefit from the advertisement in blogosphere. Thus, we can adjust the budget of advertising in blogosphere and transfer the focus of the advertisement to some other online media if the expected benefit in blogosphere is not enough.

3.1.2 Overview

At first glance, it may seem that we can predict a blogger's affinity to a cascade by analyzing the textual content of existing posts in the cascade and estimating the overlap between the content of the blogger's previous posts and cascade content. However, such *content-aware* strategy is computationally expensive and may adversely affect the accuracy of prediction for several reasons. Firstly, the content of posts are often in conversational language containing flavors of abbreviated words and local lingo. Secondly, a blog cascade may consists of posts written in different languages. Thirdly, posts may only contain multimedia objects such as pictures or video clips. Consequently, these factors make content analysis significantly challenging. Hence, we take a *content-oblivious* strategy to address this issue.

We propose a group of content-oblivious features of a blog cascade that may influence a blog's affinity to the cascade. These features can be generally categorized into two groups, namely *macroscopic* and *microscopic*. The first category refers to the features related with cascade or network, while the second one refers to those related with individual bloggers. In detail, the *macroscopic* features are associated with the overall structure of a cascade (e.g., *time elapsed* since the genesis of the cascade, *number of participants* in the cascade, and the shape of the cascade defined by the *star-likeness ratio*). On the other hand, the *microscopic* features (*number of friends* of a blogger in the cascade, *popularity of participants* in the cascade, *citing factor*, and *initiator-media link*) are related to the blogs or posts of a cascade. Note that all these features are computed by analyzing only the link structure and topology of the cascade. For each of the proposed features, we investigate how it influences a blog's affinity to the given cascade and performed a one-way analysis of variance (ANOVA) to test the significance of each feature's influence. Then we present a non-probabilistic and a probabilistic methods, namely SVM classification-based approach and *Bipartite Markov Random Field*-based (BiMRF) [86] approach, respectively, that exploit these features to predict the probability of blogs' affinity to a cascade and rank them accordingly. We did not exploit the content of the posts, our experimental results demonstrate that our prediction strategy can generate high quality results ($F1$-measure of 72.5% for SVM and 71.1% for BiMRF). In summary, the main contributions of this chapter are as follows.

- We propose an array of content-oblivious macroscopic and microscopic features that influence a blog's inclination to join a cascade. To the best of our knowledge, these features have not been studied together in the context of a blog network earlier. Further, we present different measures to calculate each feature's effect on the cascade affinity phenomenon.

- We formulate the task of predicting cascade affinity of blogs into a standard classification problem. We take two different methods, namely SVM-based and BiMRF-based classification strategies, to evaluate the probability of blogs' affinity to a particular cascade and rank them accordingly.

- We present an exhaustive evaluation of our proposed prediction and ranking methods demonstrating their effectiveness and practical significance using real-world datasets. In particular, our proposed techniques perform the best when all features except *citing factor* is used. Further, our results demonstrate that the *number of friends* feature is the most important factor affecting bloggers' inclination to join cascades.

3.2 Data Preparation

In this section, we first introduce the real-world data set we have used for our study. Then, we present our approach of cascade extraction from the data set. In the sequel, we shall use the notations shown in Table 3.1 to represent different concepts. Generally, we shall use superscript to denote a cascade identifier and subscript to denote a blog identifier.

Table 3.1: Definitions of symbols.

Symbol	Definition
b_j	blog j
c^i	cascade i
T^*	the timestamp of the last post in the data set
T^i	the timestamp when the first post appeared in c^i
$\phi^i(t)$	set of blogs that appeared in c^i before time t
ϕ^i	set of all the blogs that appeared in c^i, $\phi^i = \phi^i(T^*)$
$t^i(j)$	the timestamp when b_j joins c^i *if* $b_j \in \phi^i$; *otherwise*, $t^i(j) = T^*$
$post^i(t)$	the posts appeared in c^i before time t
$post_j(t)$	the posts appeared in blog j before time t
$s(g)$	star-likeness ratio of graph G
$G(c^i)$	shape of cascade c^i
$\mathcal{K}$	friendship threshold
$ini(c^i)$	initiator of the cascade c^i
$I(c^i)$	initiator-media link of the cascade c^i

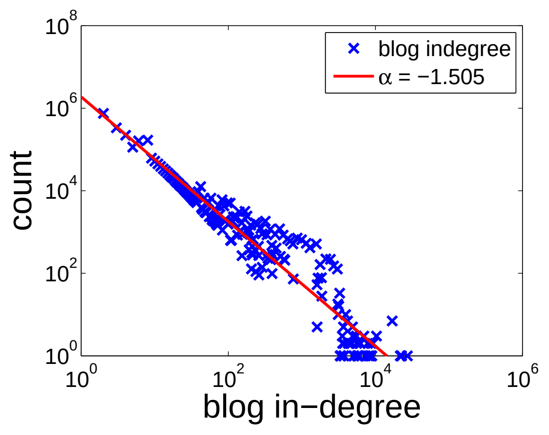
Table 3.2: Statistics of the data set.

Property	Value
Number of posts	873,469
Number of blogs	156,195
Number of blog-to-blog edges	340,124
Number of edges with weight ≥ 2	139,974
Number of cascades	7,269

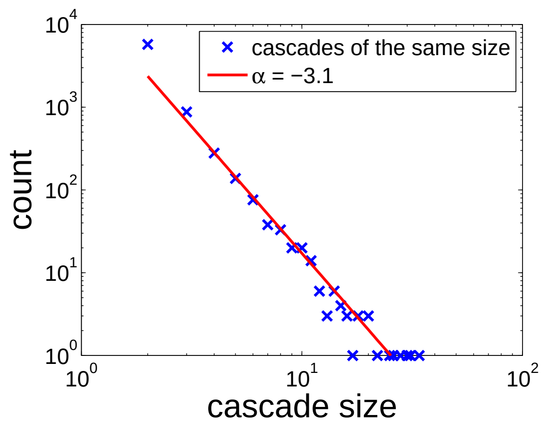
3.2.1 Dataset

We extracted our blog dataset in September, 2008 using Technorati API [2] . The data set contains blog posts published from June, 2008 to September, 2008. We first selected the group of top 100 blogs indexed by Technorati as seeds. From these seeds, we retrieved the blogs that had linked to these seeds in their posts, and then we iteratively retrieve the posts that linked

[2]http://technorati.com/developers/api



3.3.a:



3.3.b:

Figure 3.3: (a) Blog in-degree distribution (b) Cascade size distribution.

to the previous level till the sixth level which has been shown as the upper boundary size for most chain cascades [1]. From the XML collection of blogs, we can get the post-to-post relationships. Notice that a post of blog b_i linking to another post of blog b_j does not always indicate a friendship that author of b_i knows author of b_j or b_i regularly reads b_j's blog. So we additionally extracted blog-to-blog relationships with weighted edges where the *weight* of an edge from b_i to b_j indicates the number of times b_i has cited b_j's posts. Such a case, to some extent, indicates that b_i does not read b_j's blog by chance. We use this weighted graph as an indication of friends by filtering out the edges with weight less than a *friendship threshold* $\mathcal{K}$. The characteristics of the dataset is shown in Table 3.2. For each blog, the posts that do not participate in any cascade are excluded from our dataset. Figure 3.3(a) shows the in-degree distribution of blogs indexed by Technorati till September, 2008. This figure is plotted using the information extracted from our data set. It follows a power law distribution with exponent equal to -1.505 whereas in [1] this exponent is reported to be -1.7. Such a phenomenon indicates that a few blogs are more connected than the rest. It is consistent with the result of "preferential attachment" model (rich gets richer) [53].

Algorithm 1: Cascade extraction algorithm.

```
Input: A set of post-to-post relations $\mathcal{E} = \{e_1, e_2, \ldots, e_m\}$, each element is a pair of
       posts connected by a hyperlink
Output: A set of isolated cascades $\mathcal{C} = \{c^1, c^2, \ldots, c^s\}$ each of which comprised of
        connected posts
1 begin
2     initialize each cascade as a single link $\mathcal{C} \longleftarrow \mathcal{E}$;
3     while $\exists c^p, c^q$ and $c^p \cap c^q \neq \varnothing$ do
4         forall the $c^i, c^j \in \mathcal{C}$ and $c^i \neq c^j$ do
5             if $\phi^i \cap \phi^j$ then
6                 add j to i: $c^i \longleftarrow c^i, c^j$;
7                 remove j: $\mathcal{C} \longleftarrow \mathcal{C} \setminus \{c^j\}$;
```

3.2.2 Cascade Extraction

Recall that each blog participates in a cascade by writing a post which links to another post that is already in the cascade. We denote a set of cascades as $\mathcal{C} = \{c^1, c^2, \ldots, c^s\}$. The algorithm for extracting cascades from our data set is outlined in Algorithm 1.

Note that the proposed cascades extraction procedure is slightly different from the one described in [1]. Let us elaborate on this further. Consider the scenario in Figure 3.4(a), depicting blog posts and hyperlinks between them. Based on [1], each cascade should have only one initiator (top-most post). Hence, the scenario illustrated in Figure 3.4(a) have to be considered as two different cascades (have two initiators p_1 and p_2) as depicted in Figure 3.4(b). In contrast, we treat the scenario in Figure 3.4(a) as one cascade. The intuitive justification for this is as follows. Observe that the posts in Figure 3.4(a) are all linked together. That is, both p_1 and p_2 share some common posts in the conversation (*e.g.,* p_5). This may indicate that all these posts are discussing about a common topic. Hence, it makes sense to consider them as part of a single cascade instead of separating them into different ones. Note that, two posts linking together by hyperlinks indicates that their content are related with each other. In this work, we do not take into account whether they share the same sentiments or not.

The next step is to post-process the extracted cascades to eliminate the ones which have been there not more than a month till the time T^*. The set of "matured" cascades extracted after the post-processing is represented as: $\mathcal{C} = \{c^i | T^i \leq T^* - 30\}$. The number of cascades detected after filtering out the immature ones is shown in Table 3.2. The reason for post-processing the cascade set is as follows. We need to ensure that the extracted cascades can provide a robust and accurate framework for feature extraction and subsequent prediction. However, quantifying values of different features based on immature cascades (cascades which have not absorbed all potential participants) will distort the prediction accuracy of cascade affinity. Many participants may join these cascades after time T^* and consequently adversely affect the modeling of the ground truth based on the features set. Obviously, this may result in a deviation between our knowledge about the participants of these cascades and the ground truth. It is worth mentioning that it is not possible to justify the prediction performance without knowing the ground truth.

Figure 3.3(b) shows the distribution of cascade size extracted from our dataset. It is defined as the number of blogs within a cascade. The X and Y-axes represent different sizes of cascades and the number of cascades, respectively. The minimum size of cascades is defined as 2 which is the trivial case, while the maximum size of a cascade is found to be 34 in our dataset. The distribution of cascade size also follows a power law. The exponent found in our dataset is -3.1 whereas this exponent is found to be -2 in the dataset used by Leskovec et al. [1]. This deviation is primarily due to the differences between the characteristics of the two datasets and different definition of a cascade in these two approaches.

3.3 Macroscopic Features

We now present an array of content-oblivious cascade features that may influence a blog's affinity to a cascade. We classify these features into two types, namely *macroscopic* and *microscopic* features. The former refers to features that are associated with the entire cascade whereas the latter refers to features of the blogs or posts in a cascade. In this section, we

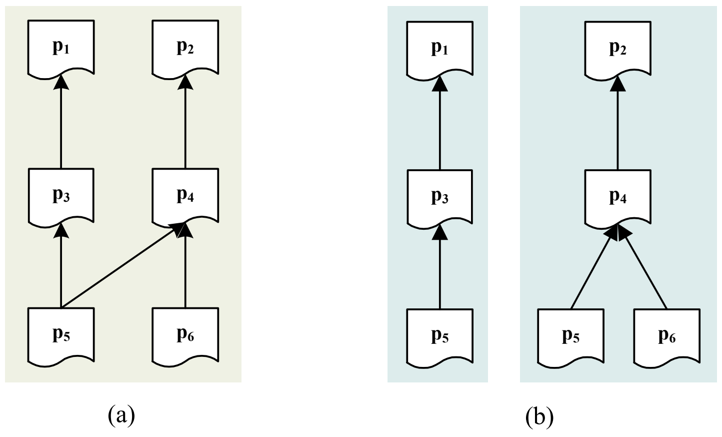


Figure 3.4: Different approaches for cascades extraction: (a) observed post-to-post relationship, it is also the cascade identified by our approach; (b) the cascades identified from (a) using the approach in [1].

begin with three macroscopic features, namely *elapsed time*, *number of participants*, and *star-likeness ratio*. In the next section, we shall elaborate on the microscopic features.

3.3.1 Elapsed Time

First we present the role of the *elapsed time*. Informally, it refers to the difference between the time a blogger joins a cascade and the cascade creation time. Formally, it is defined as follows.

Definition 3.1 *Let $t^i(j)$ be the time a blogger b_j joins a cascade c^i. Let T^i be the time of creation of c^i. Then, the elapsed time, denoted as $d^i(j)$, is defined as follows:*

$$d^i(j) \quad = \quad t^i(j) - T^i$$

We use day as the unit of elapsed time as most bloggers write posts once per day. The distribution of this feature is shown in Figure 3.5. The X-axis represents the time elapsed in days, while the Y-axis represents the number of blogs that join cascades at a specific elapsed time. Observe that 91% bloggers join a cascade during the first week. After that affinity to cascades drops almost exponentially with elapsed time. Note that the above results deviate from other types of social networks, shown in [87], where the authors found that the average number of edges attached to each node did not change much over the lifetime of the node.

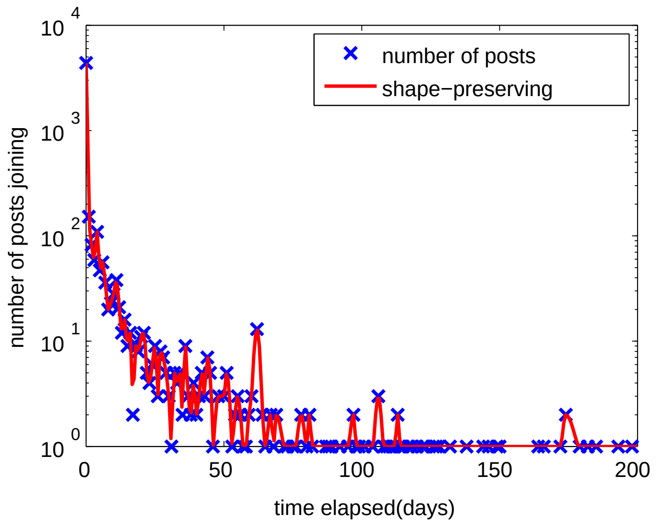


Figure 3.5: Number of posts joining versus days elapsed.

3.3.2 Number of Participants

Intuitively, a blogger may have stronger affinity to a cascade which has absorbed a lot of participants. Hence, we now conduct an analysis using *number of participants* in a cascade as a feature. We compute the probability of joining a cascade as a function of the number of participants existing in the cascade. Interestingly, we observed that in only a very small fraction (0.2%) of cascades the number of posts is more than the number blogs. It indicates that bloggers seldom re-posts in the same cascade such that the number of posts is always the same as the number of blogs in a cascade. Consequently, in the sequel we uniformly use number of blogs to represent the *size* of a cascade. The *number of participants* is formally defined as follows.

Definition 3.2 *Let $t^i(j)$ be the time when a blog b_j joins a cascade c^i. Then, the number of participants in c^i at time $t^i(j)$, denoted as $N_j(c^i)$, is defined as:*

$$N_j(c^i) = |\phi^i(t^i(j))|$$

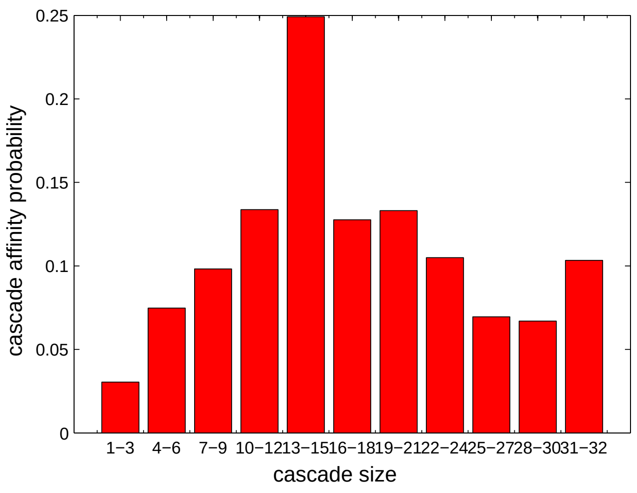


Figure 3.6: Cascade affinity probability versus cascade size.

Figure 3.6 shows the probability of joining a cascade as a function of the number of participants in that cascade. The number of blogs inside a cascade ranges from 1 to 33. The probability of joining a cascade with β participants, referred to as *cascade affinity probability* (denoted as $Pro(\beta)$), can be computed as follows.

$$Pro(\beta) \quad = \quad \frac{\sum_{c^i} |\{b_j | N_j(c^i) = \beta, b_j \in \phi^i\}|}{\sum_{c^i} |\{b_j | N_j(c^i) = \beta\}|}$$

We separate the cascades size range into 11 bins each with length 3. The height of each bar denotes the mean of the three cascade affinity probability values inside that bin. Notice that at the beginning, as the number of participants grows, the probability slightly grows, but after some point, the probability drops down. Observe the peak at the point of cascades with size $13 - 15$. It indicates that before a cascade absorbed $13 - 15$ participants, the probability for a blog to join this cascade increases. This represents the cascade initiation period where many new blogs keep on joining the cascade. However, after the number of participants in the cascade has reached a value between 13 and 15, the probability of a blog joining this cascade drops down to a stable value. This represents the stable period after a cascade has got enough attention.

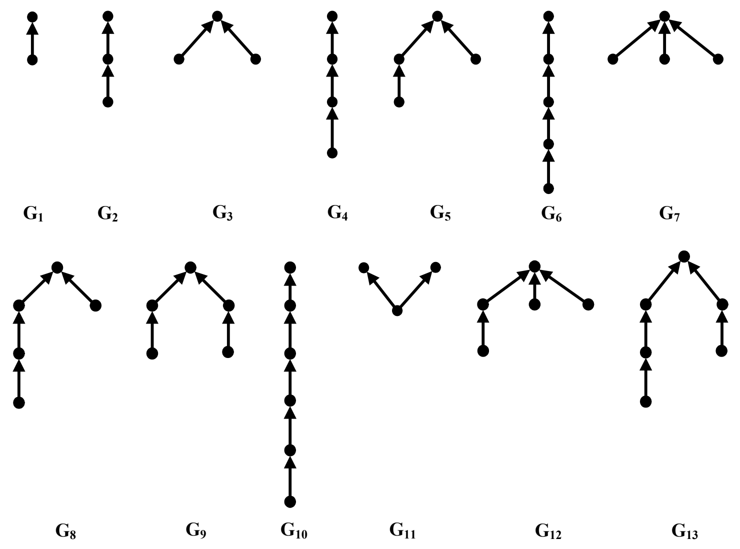


Figure 3.7: Common cascade shapes by frequency.

3.3.3 Star-likeness Ratio

Recent results showed that the type of *cascade topology* may indicate the genre of the content in a cascade [88]. We now investigate whether the topology of a cascade influences a blog's affinity to join it. Firstly, we extracted different cascade shapes in the dataset and classified the cascades into 176 different shapes. We observed that the most common shape contains only two posts. The top-13 frequent cascade shapes which appear at least 25 times in the dataset are depicted in Figure 3.7. The shapes are listed according to descending order of their frequencies (G_1 is the most frequent cascade shape). Further, the shapes can be classified into two groups, namely *chain* and *star*. Informally, a chain has only one leaf node whereas a star is an n-order shape having $n-1$ leaves. Notice that in this dataset chains appear more frequently than stars with respect to the same cascade size (*i.e.,* G_2 is more frequent than G_3 and G_{11}). Besides, shapes containing multiple-initiators are less frequent than the single-initiator ones (*i.e.,* G_3 is more frequent than G_{11}). Moreover, cascade frequency does not necessarily decrease with the increase in cascade size (*i.e.,* G_6 is more frequent than G_7).

Secondly, we investigate the probability of blogs to join a cascade by varying the cascade shapes. Formally, the *cascade shape* of c^i is defined as the following.

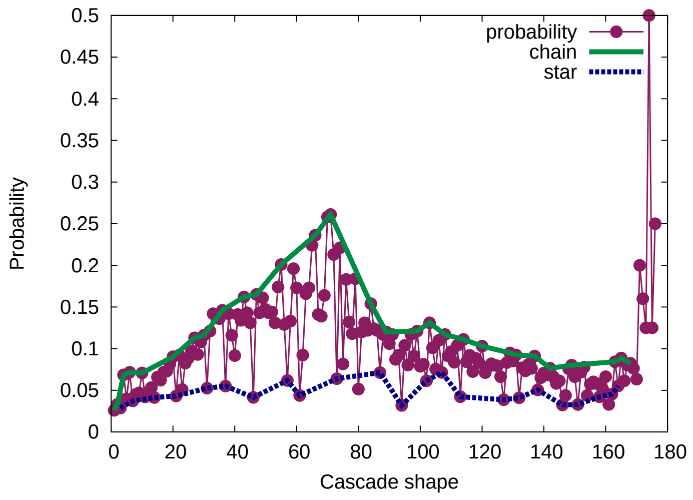


Figure 3.8: Joining probability by cascade shape.

Definition 3.3 *Let $G_1, \ldots, G_n$ denote the set of cascade shapes extracted from the dataset. If cascade c^i follows the shape G_s, then the shape of c^i, denoted as $g(c^i)$, is defined as: $g(c^i) = G_s$.*

Figure 3.8 shows the probability of joining a cascade as a function of the cascade shapes. The probability of joining a cascade of shape G_s can be computed as follows.

$$Pro_{top}(G_s) \quad = \quad \frac{\sum_{c^i} |\{b_j | g(c^i) = G_s, b_j \in \phi^i\}|}{\sum_{c^i} |\{b_j | g(c^i) = G_s\}|}$$

In general, the curve in Figure 3.8 is not informative enough to lead to any conclusion on the relationship between cascade shapes and the probability of joining a cascade. However, if we plot the probability curves for star (*i.e.,* G_3, G_7) or chain (*i.e.,* $G_1, G_2, G_4, G_6, G_{10}, \ldots$) cascades, then it is clear that chain-shaped cascades are more probable to to attract new blogs to join them compared to their star-shaped counterparts. This phenomenon may be due to the fact that new bloggers can easily see all the blogs within a chain cascade by tracking the hyperlinks one by one. In contrast, new bloggers may only see a small portion of a star-shaped cascade due to its topology. Also, observe that blogs are much more probable to join a shape with multiple roots (*i.e.,* $Pro_{top}(G_{11}) = 0.042$) compared to other shapes (*i.e.,* $Pro_{top}(G_2) = 0.033$, $Pro_{top}(G_3) = 0.029$) with the same size.

	$s(G)$	$Pro_{top}(G)$
G_1	1	**0.026**
G_2	0.5	0.037
G_3	1	**0.029**
G_4	0.333	0.069
G_5	0.667	0.040
G_6	0.25	0.072
G_7	1	**0.033**
G_8	0.5	0.045
G_9	0.5	0.047
G_{10}	0.2	0.071
G_{11}	0.25	0.042
G_{12}	0.75	0.048
G_{13}	0.4	0.053

Table 3.3: Star-likeness ratio of frequent shapes.

As discussed above, chain cascades are more probable to attract new blogs compared to their star-shaped counterparts. However, shapes other than chain or star are hard to classify and describe. Thus, we propose a new feature called *star-likeness ratio* to describe how much different a shape is from a star and utilize it in the future prediction of cascade affinity.

Definition 3.4 *Let* $root(G) = \{v | \nexists u \in V, \overrightarrow{vu} \in E\}$ *and* $leaf(G) = \{v | \nexists u \in V, \overrightarrow{uv} \in E\}$ *for the graph* $G(V,E)$. *Then the star-likeness ratio of G, denoted as* $s(G)$, *is defined as:*

$$s(G) = \frac{|leaf(G)|/|root(G)|}{|V|-1}.$$

The ratio falls within the range $(0,1]$. A shape with ratio close to 1 indicates that it is closest to star topology. All star-shaped cascades exhibit the same ratio value of 1. For example, consider the frequent shapes in Figure 3.7. The star-likeness ratio ($s(G)$) as well as the join probability $Pro_{top}(G)$ of these shapes are listed in Table 3.3. Observe that the three shapes whose star-likeness ratio are the most (*i.e.,* $s(G) = 1$) exhibit the least joining probability. This suggests that star-likeness ratio can contribute to the analysis of the cascade joining behavior.

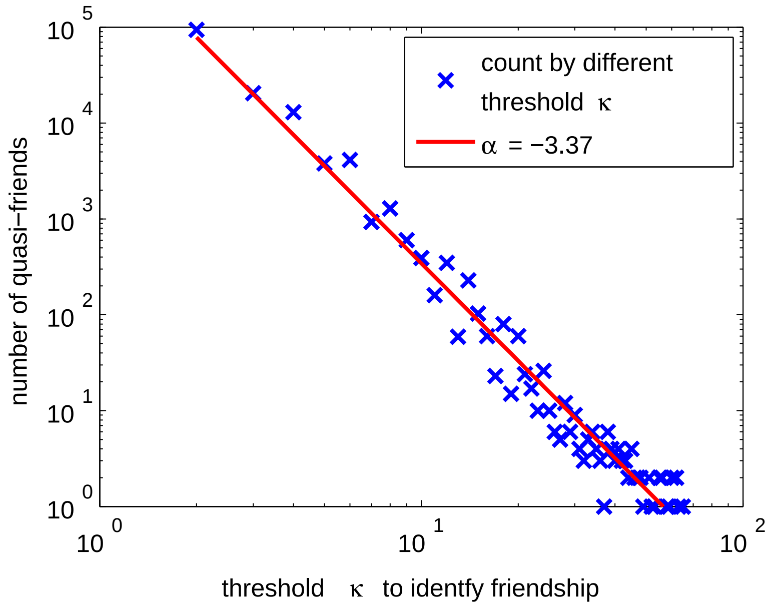


Figure 3.9: Number of quasi-friends versus $\mathcal{K}$.

3.4 Microscopic Features

In this section, we first investigate four microscopic features of blog cascades that may play important role in cascade affinity prediction, namely *number of friends*, *popularity of participants*, *citing factor*, and *initiator-media links*. We conclude this section by conducting a one-way variance analysis (ANOVA) on these macroscopic and microscopic features to quantify their significance related to cascade affinity.

3.4.1 Number of Friends

We introduce the notion of *quasi-friend* to model friendship within blogosphere based on post citings. Formally, *quasi-friend* is defined as follows.

Definition 3.5 *Given two blogs b_1 and b_2, b_1 is a quasi-friend of b_2 if and only if b_2 cites b_1's posts more than $\mathcal{K}$ times.*

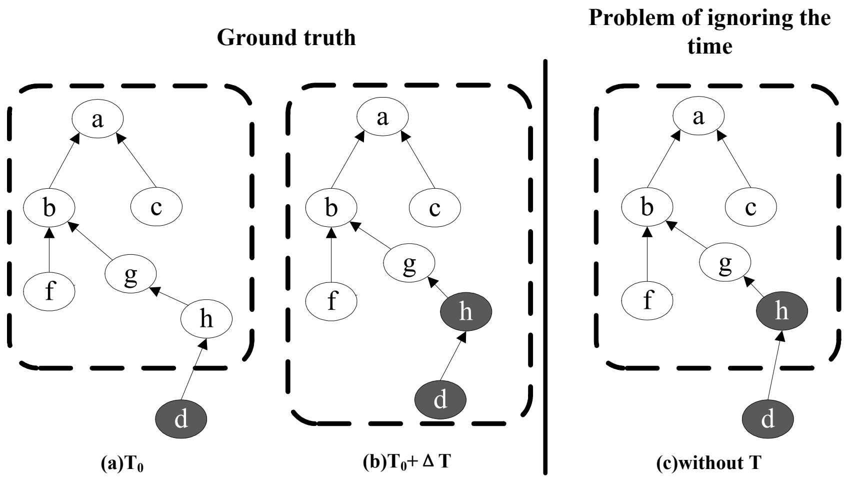


Figure 3.10: Effect of friendship creation time.

A quasi-friend indicates that b_2 probably often reads b_1's blog. This probability of frequent reading is controlled by the friendship threshold $\mathcal{K}$. Obviously, $\mathcal{K}$ will affect the number of quasi-friends discovered. As shown in Figure 3.9, $\mathcal{K}$ affects the number of quasi-friends exponentially with exponent $\alpha = -3.37$. Notice that if we set $\mathcal{K}$ to a large value then we may extract a very limited number of quasi-friends for a blog. Hence, we set $\mathcal{K}$ to 2 by default. We shall justify this value empirically in Section 6.6.1. Note that quasi-friendship is *directed*. That is, b_2 is not a quasi-friend of b_1 unless b_1 has cited b_2 more than $\mathcal{K}$ times. Given a value of $\mathcal{K}$, we denote the set of quasi-friends of a blog b_j as $F_j = \{f_1, f_2, \ldots, f_r\}$, where each element f_r is a blog.

Several recent papers have shown that personal behavior in a social network is highly affected by the person's neighbors [38, 41, 63]. Hence, the number of friends a blogger may have in a cascade is an important feature that may influence her decision to join the cascade. Naïvely, the number of friends a blogger has in a cascade can be computed at *any* time after she has joined the cascade. However, this may mislead us from the actual phenomenon as the number of friends is highly influenced by the temporal state of the cascade. Let us elaborate on this further. Consider the Figure 3.10(a). Each node is a blog and the dashed rectangle denotes a cascade at a particular time. Edges represent hyperlinks related to this cascade. Assume that

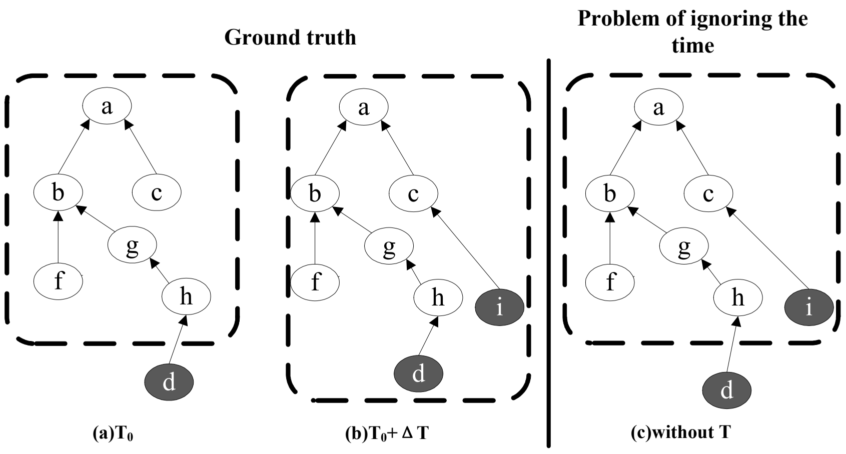


Figure 3.11: Effect of friendship creation time (contd.).

a blog d joined it at time T_0. Note that at time T_0, d did not have any friend in that cascade. We refer to T_0 as *joining time*. Now assume that at time $T_0 + \Delta T$ node h became a friend of d as shown in Figure 3.10(b). We refer to this time when a friendship is created as *friendship creation time*. Observe that the number of friends d had during joining time and friendship creation time may be different. However, if we discard these two different phenomenons, then at any time after $T_0 + \Delta T$ it may seem that d had a friend h in this community when she joined it (Figure 3.10(c)). Obviously, this is not an accurate reflection of the ground truth. Note that existing works ignore these two types of temporal features while modeling number of friends in a social network. There is another problem if we ignore the above temporal behavior. Consider the Figure 3.11(a), which represents the same scenario as depicted in Figure 3.10(a). Now assume that another blog i, who is a friend of d, joined this community at time $T_0 + \Delta T$ as shown in Figure 3.11(b). If we do not distinguish between times T_0 and $T_0 + \Delta T$, then it may seem that d had a friend i in the cascade when she joined it (Figure 3.11(c)). However, the truth is that when d joined this cascade at time T_0, she did not have any friend. Hence in our approach, we distinguish between the joining time and the friendship creation time to accurately reflect the ground truth. As we shall see in Section 6.6.1, this distinction improves the cascade affinity prediction performance significantly.

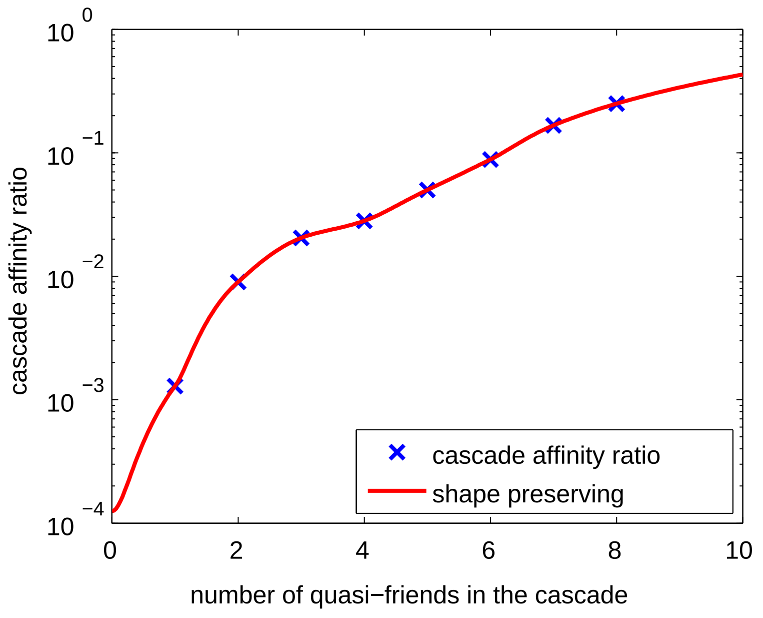


Figure 3.12: Cascade affinity ratio versus number of quasi-friends.

In our approach, we represent the set of blogs having α quasi-friends in a cascade c^i using $\Gamma^i(\alpha)$ taking into consideration the time $t^i(j)$. It is computed as follows.

$$\Gamma^i(\alpha) \quad = \quad \{b_j \| |F_j(t^i(j)) \bigcap \phi^i(t^i(j))| = \alpha\}$$

$F_j(t^i(j))$ denotes the set of blogs that became a quasi-friend of j's before time $t^i(j)$, $\phi^i(t^i(j))$ is the set of blogs that appeared in c^i before time $t^i(j)$. Note that by incorporating $t^i(j)$ in our approach, we make a contribution to address the above issues (Fig 3.10(c) and Fig 3.11(c)). Based on $\Gamma^i(\alpha)$, we define the notion of *cascade affinity ratio* with respect to the number of quasi-friends.

Definition 3.6 *Given the set of $\Gamma^i(\alpha)$, the cascade affinity ratio, denoted as P_α, is defined as:*

$$P_\alpha = \frac{\sum_i |\Gamma^i(\alpha) \bigcap \phi^i|}{\sum_i |\Gamma^i(\alpha)|}$$

We computed P_α for the whole collection of cascades, and plotted the values in Figure 3.12. The X-axis is the α value (number of quasi-friends in a cascade). The Y-axis represents the values of P_α. From the figure, we observe a diminishing return phenomenon. That is, beyond a number each additional quasi-friends in the cascade will contribute less to the probability of joining that cascade. This number is around 7 in this figure. Note that the curve depicted in the figure follows similar trend as found in other social networks [38, 63].

3.4.2 Popularity of Participants

Next we study the effect of *popularity* of cascade participants on the cascade affinity of a blogger. The idea is similar to the preferential attachment model which was first proposed in [53]. In this model, whenever a new vertex arrives in a network it attaches an edge to an existing vertex with a probability proportional to that of the old vertex's degree. Newman performed a series of analysis on the model in [89]. Leskovec et al. [87] also showed a similar pattern in some real-world data sets. Here we conduct an analysis based on this model. However, in our study when a blog joins a cascade we consider the model at the cascade-level whereas the above approaches consider it at the node level. Then, the *popularity* of a cascade c^i is the highest *rank* of the blogs in the cascade. Formally, it is defined as follows.

Definition 3.7 *Let $D(b)$ be the* rank *of a blog b. Then the popularity rank of a cascade c^i that b_j wants to join, denoted as $D_j(c^i)$, is defined as:*

$$D_j(c^i) = \min_{b \in \phi^i(t^i(j))} (D(b))$$

Note that the *rank* of each blog is based on its in-degree (indexed by Technorati). A blog having the largest in-degree has the highest rank as 1. Observe that the above definition can be intuitively explained from the social aspect. When a blogger b_j reads a post p_r she can also see other posts in the same cascade by tracing back the hyperlinks. If there is a popular blog which has a large in-degree in that cascade, then b_j will probably join this cascade. Interestingly, this effect is not so obvious in our result shown in Figure 3.13. The X-axis in the figure is the popularity rank of cascades. A cascade having lower rank means it contains a more popular

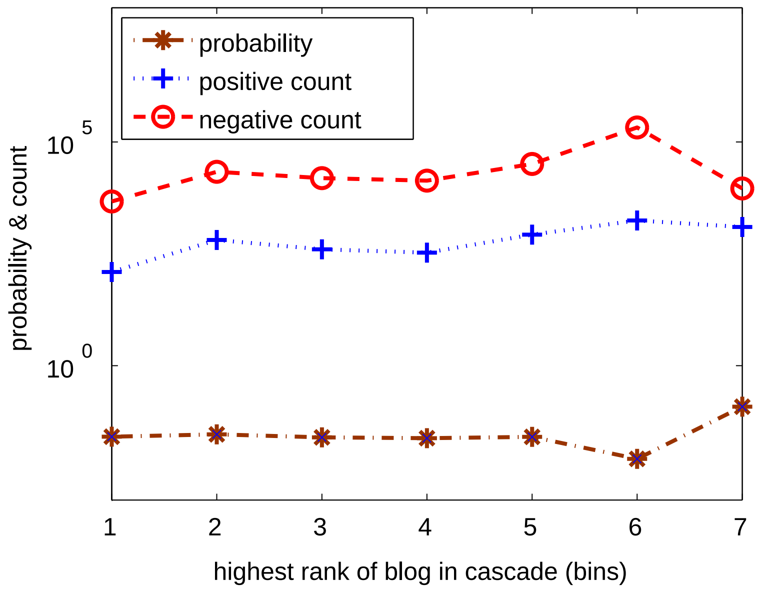


Figure 3.13: Joining probability by cascade rank.

blog. We plot the numbers of blogs that join a cascade (“positive count”) and those who do not (“negative count”) by varying the ranks. The curve labeled “probability” represents the ratio: $\frac{\text{positive count}}{\text{positive count+negative count}}$. As shown in the figure although the values along X-axis is in log-scale, the number of joined blogs in each bin do not vary much. This phenomenon indicates that a minority of cascades which have high popularity ranks influence a large number of bloggers to join.

3.4.3 Citing Factor

The features discussed above are all related to the cascade that a blog is inclined to join. Here we analyze a personal characteristics related to the joining behavior of each blogger. The reason for analyzing this feature is based on the hypothesis that a blogger b_j is more inclined to join a cascade if b_j likes to cite others’ posts.

Definition 3.8 *Let out*$(\cdot)$ *be the number of outlinks of* $\cdot$. *Then the citing factor of a blogger* b_j,

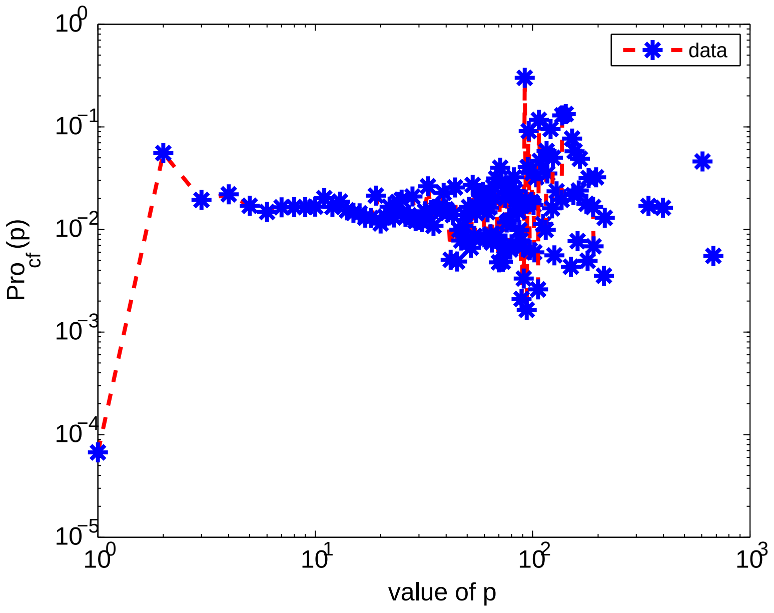


Figure 3.14: $Pro_{cf}(p)$ versus number of citations.

denoted as $H_j(c^i)$*, is defined as:*

$$H_j(c^i) = |out(post_j(t^i(j)))|$$

We can compute the probability for a blog b_j with p citations to join a cascade as follows.

$$Pro_{cf}(p) = \frac{\sum_{c^i} |\{b_j | H_j(c^i) = p, b_j \in \phi^i\}|}{\sum_{c^i} |\{b_j | H_j(c^i) = p\}|}$$

The result is shown in Figure 3.14. It is distributed almost uniformly with the variation of the number of out-links. It is evident that this feature is not very informative as far as cascade affinity is concerned.

3.4.4 Initiator-media Link

Lastly, we investigate a feature that is associated with the initiator(s) of a cascade. Recall that the initiator of a cascade is the first blogger who initiates discussion in the cascade. We conduct a series of analysis involving the cascade initiators in order to study whether there

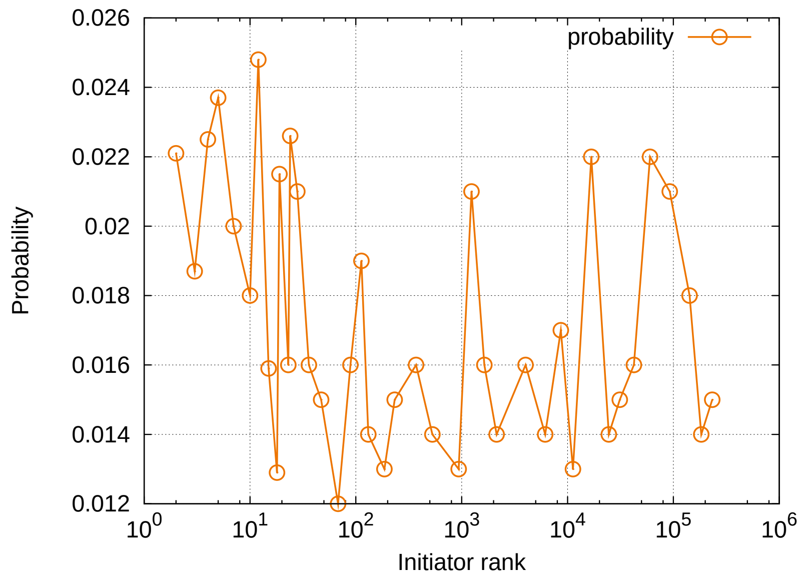


Figure 3.15: Joining probability by initiator rank.

is any correlation between the cascade affinity and the cascade initiators. First, we formally define *cascade initiator*.

Definition 3.9 *Let $c^i(B,E)$ be a cascade containing blogs $B = \{b_1, b_2, \dots, b_s\}$ with post-post links E. The initiator $ini(c^i)$ of the cascade c^i is defined as follows*

$$ini(c^i) = \{b_j | \nexists b_i \in B, \overrightarrow{b_j b_i} \in E\}.$$

Notice that it is possible to have more than one initiator in a cascade. For example, consider the cascade in Figure 3.4(a). Both p_1 and p_2 are initiators according to the above definition.

We now investigate two interesting properties of initiators of blog cascades using the Technorati dataset. We extract all the initiators for each cascade. Firstly, we investigated the correlation between the probability of a blog to join the cascade c^i and its *initiators' popularity*. Similar to Definition 3.4.2, the *popularity* of initiators of a cascade c^i is the highest *rank* of the initiators in that cascade: $\min_{b \in ini(c^i)} (D(b))$. We plot the probability of a blog to join a cascade by varying the *initiators' popularity* in Figure 3.15. Observe that there does not exist any correlation between these two factors.

Initiators	*Count*	*Avg. size of cascades*	*Ratio of join*
initiators with initiator-media links	3,782	23.32	0.11%
initiators without out-links	4,537	12.51	0.06%

Table 3.4: Initiator data.

Secondly, we examine a property of the initiators related to their out-links to other media resources. Initiators in a cascade are the posts that do not reference any other post in that cascade. However, they may reference non-blog media sources such as Flickr, Youtube, etc. We refer to these links as *initiator-media links*. Formally, it is defined as follows.

Definition 3.10 *Let c^i denote the cascade. Then, the initiator-media link of c^i, denoted as $I(c^i)$, is defined as:*

$$I(c^i) = \begin{cases} 1 & \text{if } \exists b \in ini(c^i),\ b \text{ hyperlinks to non-blog URLs} \\ 0 & \text{otherwise.} \end{cases}$$

Table 3.4 reports the statistics of initiators that link to these media resources against those which do not. Observe that the average size of cascades containing initiator-media links is larger than those which do not have such link. It means that the initiators who reference other media resources are probable to generate larger cascades than the ones that do not. This phenomenon suggests that bloggers are more inclined to write post on a topic when they have found related resources from many different media.

Thus, it indicates that bloggers are more probable to join the cascades whose initiators referenced other media resources. Hence, we propose to use this property as another feature to predict the cascade affinity. For each cascade, we first identify the initiators within it. After that, we test whether the initiators have hyperlinks to web pages that belong to non-blog domains.

3.4.5 ANOVA Test

In this section, we conduct a one-way variance analysis (ANOVA) on each of the aforementioned macroscopic and microscopic features to quantify their significance related to cascade

	Feature Name	*F*	*p-value*
Macro Features	Time elapsed	6.88	$\ll 0.001$
	Number of participants	4.36	$\ll 0.001$
	star-likeness ratio	1.66	0.024
Micro Features	Number of friends	2.85	0.017
	Popularity of participants	1.50	0.029
	Citing factor	0.77	0.968
	Initiator-media link	1.38	0.034

Table 3.5: ANOVA test on cascade features.

affinity. For each feature, we compare the values between blogs which finally joined a cascade and those did not using the one-way analysis of variance (ANOVA) to test whether the difference is really caused by the feature values or just by noise in the data. The F and *p*-values of each feature is shown in Table 3.5. The result shows that the *p*-value for citation factor is 0.968 while other features are all less than 0.05. It indicates that the different values of citation factor in both groups should only be considered as noise. The remaining six cascade features are all significant for predicting cascade affinity of a blogger.

3.5 Cascade Affinity Prediction

In this section, we describe how the features discussed in previous sections can be exploited to predict bloggers who may join a cascade. The prediction involves two steps, namely *candidate blog extraction* and *cascade joining prediction*. We elaborate on these steps in turn.

3.5.1 Candidate Blog Extraction

For a given cascade, all blogs in the blogosphere are potential blogs that may join the cascade in the future. Nevertheless, many of these potential blogs have no interaction (*e.g.,* read the posts) with the blogs/posts already in the cascade and are unlikely to join the cascade. We therefore only consider a much smaller set of *candidate blogs* that are likely to read one or more posts in the cascade. The candidate blogs are those that have at least one quasi-friend in

Algorithm 2: Candidate blog extraction algorithm.

```
  Input: cascade set $\mathcal{C} = \{c_1, c_2, \ldots, c_s\}$ extracted from the data set
  Output: candidates $\Delta^i$ for each cascade $c^i$
1 begin
2     foreach cascade $c^i \in \mathcal{C}$ do
3         foreach blog $b_j \in \phi^i$ do
4             $\Delta^i(j) = \{r | b_j \in F_r(t^i(j))\}$;
5             $\Delta^i = \Delta^i \bigcup \Delta^i(j)$;
```

the given cascade. Formally, for a given cascade c^i, the candidate blogs $cand(c^i)$ that may join c^i is given by the following equation.

$$cand(c^i) = \{j | F_j \cap \phi^i \neq \varnothing\} \qquad \text{(Eq. 3.1)}$$

The algorithm for extracting candidate blogs is outlined in Algorithm 2. Recall that quasi-friend is defined based on the number of times (*i.e.,* $\mathcal{K}$) a blog cites posts from another blog. Hence, the number of candidate blogs extracted for a given cascade naturally depends on the threshold $\mathcal{K}$. In our experiments, we set $\mathcal{K} = 2$ by default.

For all cascades in our dataset, there are 312,414 candidate blogs extracted by Algorithm 2. On average, 43 candidates are extracted for each cascade. Naturally, the number of candidate blogs increases along the number of participants in a cascade. Particularly, for a cascade having fewer than 10 participants, there are 39 candidate blogs on average; for a cascade having $11-20$ participants, this value increases to 64 candidates on average; for a cascade having more than 20 participants, there are 81 candidates on average. From the numbers reported, candidate blog extraction greatly reduces the number of blogs to be considered in the prediction with respect to the total number of blogs in our data set. As an evaluation of candidate extraction, Table 3.6 shows 76.1% blogs that join a cascade have at least a quasi-friend in it when we set $\mathcal{K} = 2$.

3.5.2 SVM-based Cascade Affinity Prediction

We now present two techniques that exploits the macroscopic and microscopic features to predict blogs that may join a cascade. One takes a *non-probabilistic* approach whereas the other is *probabilistic* in nature. In this subsection, we present the former approach first. We discuss the probabilistic approach in the next subsection. In Section 3.6, we shall empirically compare these two strategies.

The prediction task can be naturally formulated as a binary classification task. Many existing classifiers (*e.g.,* Naïve Bayes, *k*-Nearest Neighbors, and Support Vector Machines) indeed return a category relevance score for each data instance to be classified indicating its likelihood of belonging to a pre-defined category. We adopt a non-probabilistic binary classifier Support Vector Machines (SVM) [90] due to its promising results reported in many data mining/machine learning tasks. SVM models all the samples including both positive and negative ones as points in high dimensional space. The training of SVM learns a hyperplane in the space. The hyperplane learned from SVM model should be able to separate the positive training examples from the negative ones with the largest margin.

Formally, the training data is represented as a set of points in a 7-dimensional space: $\mathcal{D} = \{(\mathbf{x}_i, y_i) | \mathbf{x}_i \in \mathbb{R}^7, y_i \in \{-1, 1\}\}$. y_i is either 1 or -1, indicating the class to which the point belongs. $\mathbf{x}_i$ is a 7-dimensional vector. Our target is to find a hyperplane that divides the points having $y_i = 1$ from those having $y_i = -1$ with the maximal margin. Actually, any hyperplane can be written as the set of points $\mathbf{x}$ that satisfies the following equation:

$$\mathbf{w} \cdot \mathbf{x} - b = 0$$

where $\mathbf{w}$ is normal vector and perpendicular to the hyperplane. The vector $\mathbf{w}$ and a parameter b to be learned from the training data by minimizing the function:

$$\mathbf{w}^\top \cdot \mathbf{w}$$

subject to

$$y_i(\mathbf{w} \cdot \mathbf{x}_i - b) \geq 1.$$

In order to learn an SVM classifier, those candidate blogs that eventually joined and did not join the target cascades were used as positive and negative examples, respectively. Moreover, all the candidates extracted using Algorithm 2 are formulated as vectors in order to fit in the model. For example, a candidate sample b_j is represented as 7-dimensional vector:

$$\overrightarrow{b_j} = [b_{j1}, b_{j2}, \ldots, b_{j7}]^\top$$

Within the vector, each entry is the value of one of the seven macroscopic and microscopic features discussed in Sections 3.3 and 3.4.

Given the learned model, we compute a score for an unlabeled object b_j using its decision function:

$$f(b_j) = \mathbf{w} \cdot b_j - b.$$

In our setting, a larger $f(b_j)$ indicates more likelihood of b_j joining the target cascade.

3.5.3 Bipartite Markov Random Field-based (BiMRF) Cascade Affinity Prediction

Other than SVM, we then adopted a probabilistic approach to predict the likelihood of joining a cascade which is based on *Bipartite Markov Random Field* (BiMRF). BiMRF [86] models the group joining behavior as a bipartite graph where the vertices at one side are associated with the variables $B = \{b_i\}_{i=1}^N$ which represent users, and the vertices at the other side are associated with variables $C = \{c^j\}_{j=1}^M$ which represent cascades. The advantage of using the BiMRF model in this problem is that it can explicitly incorporate the relationship between bloggers and cascades. Moreover, it has been proved to be more effective than other approaches in modeling the group joining behavior [86]. Similar to SVM-based approach, based on the observed value of each feature, we study the joining behavior at different time step independently using BiMRF model. In the model, each user is a 7-dimensional feature vector $b_i = [b_{i1}, b_{i2}, \ldots, b_{i7}]^\top$, the values of which may change over time. Let O denote all the observations, including users and their features, cascades and their features as well as the connections of users. Let $E = \{E_{ij}^t : 1 \leq i \leq N, 1 \leq j \leq M \text{ and } 1 \leq t \leq T\}$ be a set of random variables where $e_{ij} = 1$ if the user b_i joins cascade c^j at time t; otherwise it is 0. Let $\{e\}$ denote an instance of E. Then given the observations, BiMRF defines a conditional distribution as follows:

$$p(\{e_t\}|O) = \frac{1}{Z(w)} \exp(\sum_{k=1}^{K} \omega_k f_k(\{e_t\}, O))$$

where f_k are feature functions and ω_k are their weights which will be learned. As BiMRF treats the joining behavior at different time snapshots independently given the observed features, $p(\{e\}|O) = \prod_{t=1}^{T} p(\{e\}|O)$.

Formally, the dataset is a pairing of observations and joining behaviors (*i.e.,* $\mathcal{D} = \{\langle\{e\}, O\rangle\}$). The best model to fit the data is the one with the maximum conditional likelihood: $\mathcal{L} = \log p(\{e\}|O)$. We define feature functions to compute the feature values $f_k(\{e\}, O)$ in the above equation. For example, the feature function to compute the feature value of *number of participants* is as follows.

$$f_{nop}(e_{ij}^t = 1, c^i, b_j, t) = N_j(c^i).$$

Thus, the optimized ω_k is learned by maximizing the likelihood:

$$\mathcal{L} = \log p(\{e\}|O).$$

In line with [86], the optimization is achieved using L-BFGS (Limited Memory Broyden-Fletcher-Goldfarb-Shanno) algorithm [91].

With the weights ω_k learned from the training data, the probability that user b_j joins cascade c^i at time t is given by computing the marginal probability

$$p(e_{ij}^t = 1|O).$$

3.6 Experiments

3.6.1 Experimental Setting

For both prediction models, we conducted experiments on our data set using 5-fold cross validation to evaluate the effectiveness of the features in predicting cascade affinity of candidate blogs. That is, the data set was randomly partitioned into 5 parts and in each evaluation, 4 parts were used as training data and the remaining part was used as test data. The results reported are averaged over the 5 runs.

The commonly used performance evaluation measures in classification tasks are *precision*, *recall* and F_1. Precision, denoted by *Pr*, is the percentage of blogs that eventually joined the target cascade among all blogs predicted to be joining. Recall, denoted by *Re*, is the percentage of the correct predictions among all blogs that eventually joined the target cascade. Note that, recall is computed with respect to all blogs that finally joined the target cascade regardless of whether the blogs are identified as candidate blogs or otherwise. $F_1 = \frac{2 \times Pr \times Re}{Pr + Re}$ is the harmonic mean of precision and recall. However, both precision and recall are threshold-dependent. A higher threshold leads to higher precision but lower recall. In our experiments, we are more interested in the effectiveness of the features in ranking the candidate blogs according to the likelihood of joining the target cascade. We therefore adopted the area under Precision-Recall curve (AUC-PR) as the evaluation metric.

Value of $\mathcal{K}$	*Candidate size (max. recall)*	*Highest F_1-measure*	
		SVM	BiMRF
$\mathcal{K}$=1	946,329 (0.916)	0.707	0.702
$\mathcal{K}$=2	312,414 (0.761)	**0.725**	**0.711**
$\mathcal{K}$=3	80,482 (0.242)	0.227	0.227

Table 3.6: Effect of different values of $\mathcal{K}$.

3.6.2 Experimental Results

Justification of candidate set. Recall that the number of candidate blogs is affected by the parameter $\mathcal{K}$. As $\mathcal{K}$ increases, the number of quasi-friends identified decreases. Consequently, the candidate blog set shrinks. As a result, the maximum recall decreases, but the prediction performance may not. To determine the optimum value for $\mathcal{K}$, we conducted the prediction using different values of $\mathcal{K}$. Table 3.6 shows the sizes of candidate blog sets for different $\mathcal{K}$ as well as the highest F_1-measures achieved by selecting the best thresholds in both models. Observe that in both models the best F_1-measures are achieved at $\mathcal{K} = 2$. Hence, in the subsequent experiments we shall set $\mathcal{K} = 2$.

Comparison of feature sets. Recall that we have identified seven features for cascade affinity prediction, namely *number of friends*, *popularity of participants*, *number of participants*, *citing factor*, *elapsed time*, *star-likeness ratio* and *initiator-media link.* To evaluate the effectiveness of these features, we conduct 8 sets of experiments. The first set of experiments uses all seven features for prediction. This feature set is denoted by "ALL" in Table 3.7. In each of the following seven experiments, one feature is removed. For instance, "A-NF" denotes that the feature *number of friends* is removed and the remaining six features are used for prediction. In Table 3.7, a '✓' indicates that the feature is used and '-' otherwise.

The prediction performances measured by AUC-PR are reported in the last two rows in Table 3.7. Using all the seven features, the prediction achieves AUC-PR of 0.615 for SVM and 0.610 for BiMRF. We can make the following observations:

Features/AUC-PR/Feature set	ALL	A-NF	A-PP	A-NP	A-CF	A-ET	A-IP	A-SR
Number of friends	✓	-	✓	✓	✓	✓	✓	✓
Popularity of participants	✓	✓	-	✓	✓	✓	✓	✓
Number of participants	✓	✓	✓	-	✓	✓	✓	✓
Citing factor	✓	✓	✓	✓	-	✓	✓	✓
Elapsed time	✓	✓	✓	✓	✓	-	✓	✓
Initiator-media link	✓	✓	✓	✓	✓	✓	-	✓
star-likeness ratio	✓	✓	✓	✓	✓	✓	✓	-
AUC-PR (SVM)	0.615	0.066	0.588	0.604	**0.625**	0.604	0.592	0.595
AUC-PR (BiMRF)	0.610	0.055	0.588	0.599	**0.618**	0.587	0.587	0.589

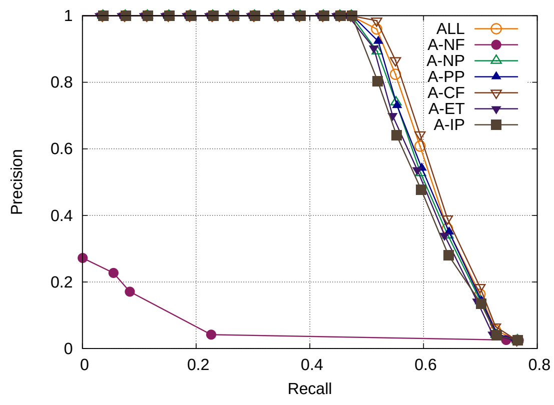
Table 3.7: Feature set notations and prediction performance in AUC-PR

- Removal of *number of friends* resulted in significant drop in prediction performance to 0.046 in both models indicating that *number of friends* is the most important factor that affects a blogger's cascade affinity.

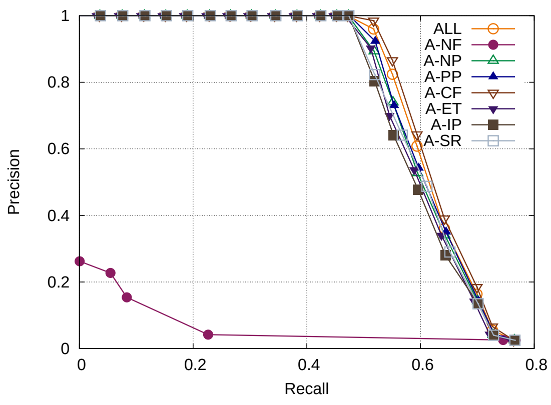
- Removal of either *popularity of participants*, *number of participates*, *star-likeness ratio*, *elapsed time* or *initiator-media link* led to a small performance degradation. These five features indeed contribute to the cascade affinity modeling.

- An interesting observation is that removal of the *citing factor* in both models led to better AUC-PR than using all the seven features. This result clearly indicates that the *citing factor* introduces noise in the prediction, which is consistent with our ANOVA test results reported in Section 3.4.5. The remaining six features, *number of friends*, *popularity of participants*, *number of participates*, *elapsed time*, *star-likeness ratio* and *initiator-media link* achieve the best performance.

- We conduct additional two experiments using only macroscopic or microscopic features.



3.16.a: SVM



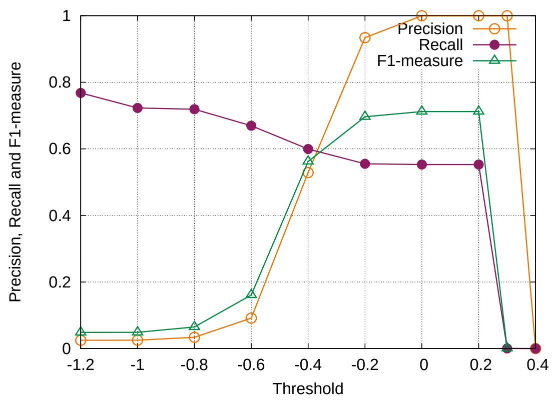
3.16.b: BiMRF

Figure 3.16: Precision-Recall Curves for different features.

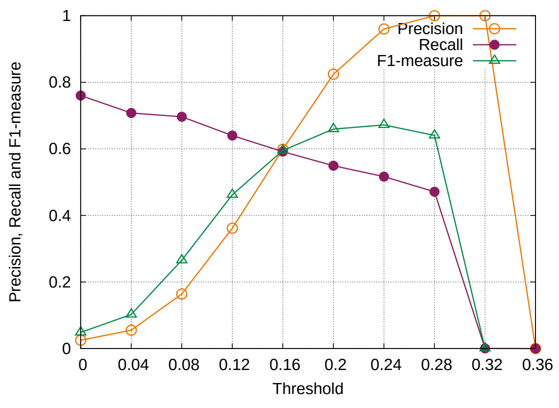
Microscopic features achieve AUC-PR of 0.533 for SVM (0.519 for BiMRF) whereas macroscopic ones only exhibit AUC-PR of 0.064 for SVM (0.059 for BiMRF). This is because the *number of friends* is one of the microscopic features.

Figure 3.16 plots the Precision-Recall curves of using eight different feature sets. Under the SVM model, all the seven runs (except for "A-NF") achieve almost perfect precision before recall reached 0.57. Sharp drop of precision is then observed along with the increase of recall. In contrast, all the seven runs of BiMRF model (except for "A-NF") achieve almost perfect precision before recall reached 0.48. As the recall increases, precision in BiMRF drops down more smoothly than that of SVM model. However, SVM and BiMRF show almost the same AUC-PR. Thus, both models can be applied for cascade affinity prediction.

Figure 3.17 shows the precision, recall and F_1-measure by varying the threshold for the feature set "A-CF", which has the best prediction performance among all the approaches. *Citing factor* introduces noise in the prediction may be caused by the following reasons. Firstly, bloggers may not always post on the same blog site such that we are not able to completely acquire their historical postings. Secondly, citing factor may not be independent with some

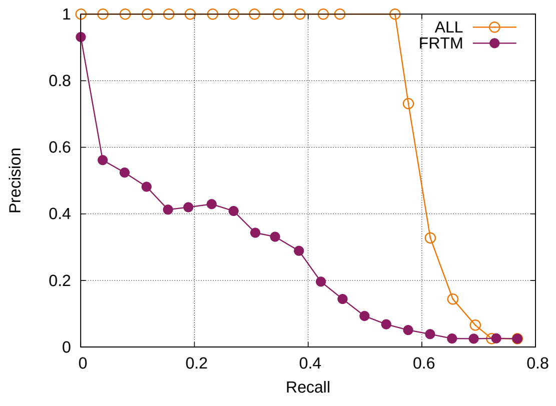


3.17.a: SVM

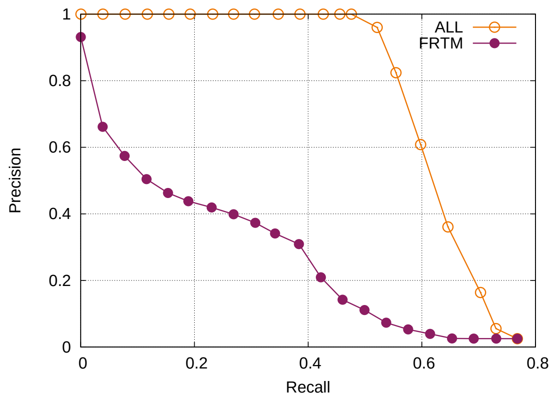


3.17.b: BiMRF

Figure 3.17: Precision, Recall and $F1$-measure for “A-CF”.



3.18.a: SVM



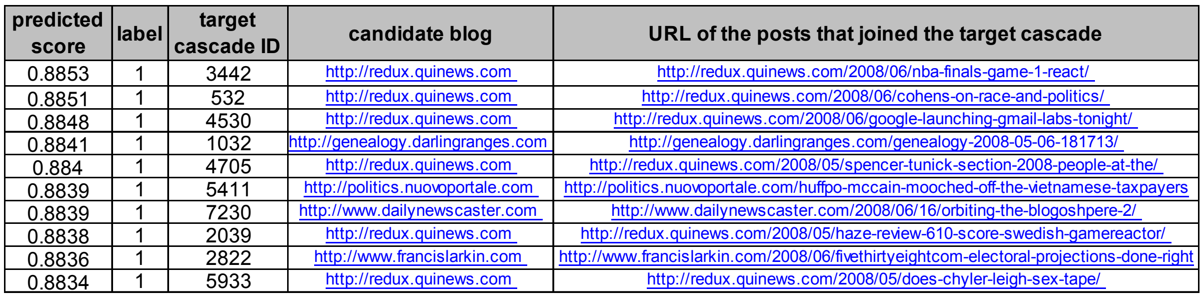
3.18.b: BiMRF

Figure 3.18: Significance of time for modeling number of friends.

other feature examined (*i.e.,* number of friends) which causes their contributions inaccurately evaluated.

Both precision and recall of BiMRF model change more smoothly than those of SVM model as the thresholds increase, indicating that SVM model results in a clearer margin between the score of positive samples and negative ones.

Significance of time for modeling number of friends. Recall that in Section 3.4.1, we illus-



predicted score	label	target cascade ID	candidate blog	URL of the posts that joined the target cascade
0.8853	1	3442	http://redux.quinews.com	http://redux.quinews.com/2008/06/nba-finals-game-1-react/
0.8851	1	532	http://redux.quinews.com	http://redux.quinews.com/2008/06/cohens-on-race-and-politics/
0.8848	1	4530	http://redux.quinews.com	http://redux.quinews.com/2008/06/google-launching-gmail-labs-tonight/
0.8841	1	1032	http://genealogy.darlingranges.com	http://genealogy.darlingranges.com/genealogy-2008-05-06-181713/
0.884	1	4705	http://redux.quinews.com	http://redux.quinews.com/2008/05/spencer-tunick-section-2008-people-at-the/
0.8839	1	5411	http://politics.nuovoportale.com	http://politics.nuovoportale.com/huffpo-mccain-mooched-off-the-vietnamese-taxpayers
0.8839	1	7230	http://www.dailynewscaster.com	http://www.dailynewscaster.com/2008/06/16/orbiting-the-blogoshpere-2/
0.8838	1	2039	http://redux.quinews.com	http://redux.quinews.com/2008/05/haze-review-610-score-swedish-gamereactor/
0.8836	1	2822	http://www.francislarkin.com	http://www.francislarkin.com/2008/06/fivethirtyeightcom-electoral-projections-done-right
0.8834	1	5933	http://redux.quinews.com	http://redux.quinews.com/2008/05/does-chyler-leigh-sex-tape/

Figure 3.19: Top 10 candidates that are most probable to join a cascade (SVM).

trated the significance of time in modeling the number of friends. To justify the goodness of our solution, we compare it with the approach that ignores time. Specifically, if we discard the temporal issue in modeling quasi-friends then the definition of $\Gamma^i(\alpha)$ (the set of blogs having α friends in cascade c^i) is modified as follows.

$$\Gamma^i(\alpha) = \{b_j \| F_j(T^*) \bigcap \phi^i| = \alpha\}$$

We update the *number of friends* feature in each candidate vector using the above formula. Using the updated feature vectors, we perform the prediction again. The performance of ignoring the time in *quasi-friend* identification shows a small AUC-PR 0.211 (0.216 for BiMRF) whereas our proposed solution achieves 0.615 (0.610 for BiMRF). The comparison between the Precision-Recall curve of this approach and our proposed solution is shown in Figure 3.18. Both curves use all the seven features. "FRTM" represents the approach that discards the temporal aspects in *number of friends*. It is clear that time is an important factor in *quasi-friend* identification as it achieves significantly better prediction compared to the approach that ignores time.

Prediction of top-k bloggers. To study the prediction accuracy of top-k blogs that are inclined to join a cascade, we compute the precision of our approach to retrieve top-k bloggers ranked based on the predicted scores. Specifically, for each cascade c^i having more than k positive samples, we generate the top-k predicted blogs and compute the *precision* as follows: $Pr^i(k) =$

predicted score	label	target cascade ID	candidate blog	URL of the posts that joined the target cascade
0.4314	1	5411	http://politics.nuovoportale.com	http://politics.nuovoportale.com/huffpo-mccain-mooched-off-the-vietnamese-taxpayers
0.4281	1	4530	http://redux.quinews.com	http://redux.quinews.com/2008/06/google-launching-gmail-labs-tonight/
0.4218	1	3442	http://redux.quinews.com	http://redux.quinews.com/2008/06/nba-finals-game-1-react/
0.4202	1	532	http://redux.quinews.com	http://redux.quinews.com/2008/06/cohens-on-race-and-politics/
0.4202	1	7230	http://www.dailynewscaster.com	http://www.dailynewscaster.com/2008/06/16/orbiting-the-blogoshpere-2/
0.4185	1	2822	http://www.francislarkin.com	http://www.francislarkin.com/2008/06/fivethirtyeightcom-electoral-projections-done-right
0.4164	1	1032	http://genealogy.darlingranges.com	http://genealogy.darlingranges.com/genealogy-2008-05-06-181713/
0.4153	1	3144	http://redux.quinews.com	http://redux.quinews.com/2008/06/jim-johnson-obama/
0.3914	1	2039	http://redux.quinews.com	http://redux.quinews.com/2008/05/haze-review-610-score-swedish-gamereactor/
0.3814	1	4621	http://redux.quinews.com	http://redux.quinews.com/2008/05/free-battlestar-galactica-episodes/

Figure 3.20: Top 10 candidates that are most probable to join a cascade (BiMRF).

k	1	2	3	4	5
$Pr_{avg}(k)$ (SVM)	**0.970**	0.783	0.722	0.734	0.763
$Pr_{avg}(k)$ (BiMRF)	**0.970**	0.762	0.716	0.722	0.744

Table 3.8: Average precision versus top-k.

$\frac{\#\text{true positive}}{k}$. Then for a given k, we compute the *average precision*, denoted as $Pr_{avg}(k)$, using the following formula.

$$Pr_{avg}(k) = \frac{\sum_{|\phi^i|\geq k} Pr^i(k)}{\left|\{c^i \big| |\phi^i| \geq k\}\right|}$$

Table 3.8 shows average precision values for different k values highlighting the goodness of our approach. Note that $Pr_{avg}(k)$ may not monotonically decrease with increasing k as the number of cascades in the denominator depends on k.

Figures 3.19 and 3.20 show the top-10 candidate blogs over entire cascades collection. If the candidate is a positive sample, we also show the corresponding URL of the post that joins the target cascade.

Using only the *number of friends* as feature. As mentioned above, the *number of friends* is a key feature that affects a blogger's cascade affinity. To further validate this, we conducted another experiment using only the *number of friends* as feature. The AUC-PR in this case is 0.445 for SVM and 0.434 for BiMRF.

In fact, if we list the top-10 candidate blogs (similar to Figures 3.19 and Figures 3.20) using only the *number of friends* in SVM or BiMRF, we can also get 10 positive samples which

k	1	2	3	4	5
$Pr_{avg}(k)$ (SVM)	**0.643**	0.561	0.597	0.571	0.573
$Pr_{avg}(k)$ (BiMRF)	**0.612**	0.515	0.535	0.564	0.558

Table 3.9: Average precision versus top-k using only *number of friends*.

eventually join the target cascades. However, when we compute the average precision values for different k values (similar to Table 3.8), the aforementioned approach shows poor results as reported in Table 3.9. It indicates that the approach using only the *number of friends* as feature is not superior to the proposed technique that uses an array of features. The former can only find very limited number of candidate blogs which are most likely to join target cascades but fails to demonstrate enough precision in order to predict the affinity in each individual cascade.

Comparison of the performance between SVM and BiMRF model. We compare the performances of the SVM and BiMRF models using the prediction results shown in Table 3.7 and Figure 3.17. We can make the following observations.

- In both models, removal of the *citing factor* perform the best, then is the approach using all the features. However, the removal of either *elapsed time* or *initiator-media link* does not perform better than the removal of *popularity of participants* in BiMRF, which is not the case in SVM.

- In general, both SVM and BiMRF models show satisfactory prediction results with AUC-PR over 0.61 when using the best feature set "A-CF". However, SVM performs slightly better than BiMRF model for all the other feature sets.

- The precision and recall curves by varying the threshold of both SVM and BiMRF model almost exhibit the same shape, except that the curves of BiMRF model is smoother than those of SVM. It suggests that SVM model tends to produce clearer margin between positive class and negative one.

- Both SVM and BiMRF models exhibit the best average precision with 0.970 when predicting the top-1 blogger according to Table 3.8. In addition, the average precision of top-k ($k = 2, \ldots, 5$) of both methods do not vary much. Moreover, 8 out of 10 records in Figure 3.20 also appear in Figure 3.19, indicating that the results of both models do not vary much.

3.7 Summary

In this chapter, we analyzed a large publicly available collections of blog information to investigate bloggers' behavior and interaction with blog cascades. We have identified in total seven macroscopic and microscopic features, namely number of friends, popularity of participants, number of participants, time elapsed since the genesis of the cascade, star-likeness ratio, initiator-media link and citing factor of the blog, that may play important role in predicting blog cascade affinity so as to identify most easily influenced bloggers. Such bloggers play important role in several real-world applications. Note that our proposed features are derived from structural information of the cascades without any content analysis of posts/blogs. We performed ANOVA test on these features and showed that all of them, except citation factor, have significant impact on cascade affinity. The cascade affinity prediction is then formulated as a classification task and a non-probabilistic (SVM-based) and a probabilistic (BiMRF classifier) methods are employed. Using the prediction scores from the SVM-based approach or the conditional probability from BiMRF, the candidate blogs can be ranked according to their probability of joining a cascade. We have evaluated different combinations of the features and our results on cascade affinity prediction is consistent with the ANOVA test. In general, SVM model performs slightly better than BiMRF model in most of the feature sets. Moreover, SVM model tends to generate clearer margin between positive class and negative one. The six features that have significant impact on cascade affinity achieved the best prediction accuracy of

0.625 (0.618 for BiMRF) measured by AUC-PR. Our experimental results also showed that the number of friends plays a significant role in blog cascade affinity prediction.

Chapter 4

CASINO: Towards Conformity-aware Social Influence Analysis

In the previous chapter, we present a model to rank the object nodes according to their probability to join a blog cascade. In this chapter, we shall present a general model to evaluate the inclination of those object nodes to be influenced in arbitrary social networks. The rest of this chapter is organized as follows. We first introduce the background of social influence analysis and motivate the research in conformity-aware social influence analysis in Section 4.1. In Section 4.2 we present the concept of influence and conformity indices. After that, we discuss the computation algorithm of both indices in Section 4.3. The experimental result in discussed in Section 4.4. Finally, we conclude the work in this chapter in Section 4.5. A preliminary version of this chapter has been published in [92].

4.1 Introduction

The goal of social influence analysis is to study individuals' influence by analyzing the social interactions between people. Recently, it has attracted tremendous research interest due to its role in governing the interactions in social networks as well as understanding the spread patterns of social influence. By identifying the "influentials" in a social network, users may be able to maximize the influence of a piece of information [4, 42, 64]. Informally, influentials

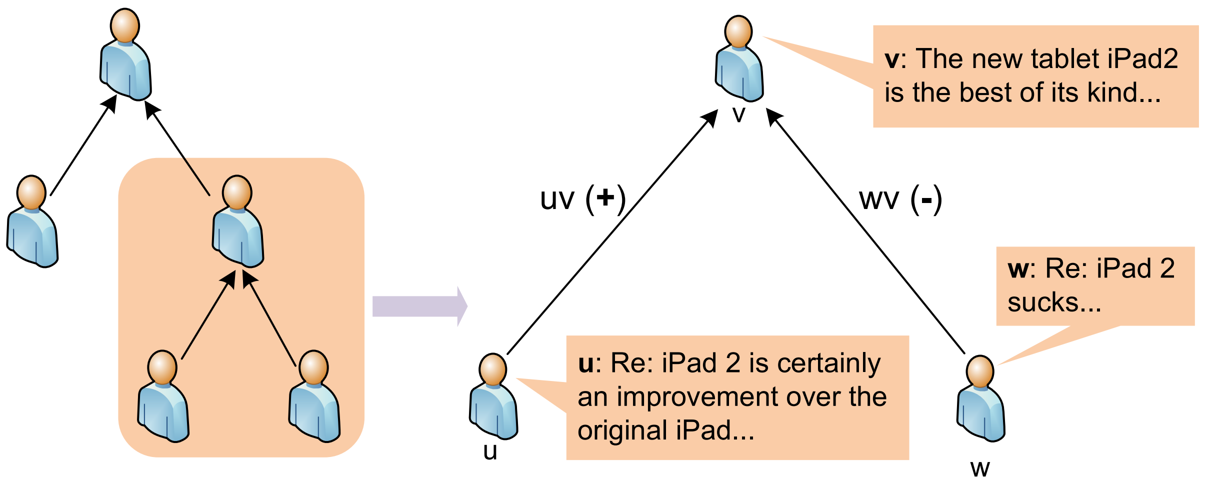

v: The new tablet iPad2 is the best of its kind...
v
uv (+)
wv (-)
w: Re: iPad 2 sucks...
u: Re: iPad 2 is certainly an improvement over the original iPad...
u
w

Figure 4.1: Social influence propagation scenario.

are those individuals whose opinions or advices are often accepted and supported by others. For instance, Domingos and Richardson [64, 66] are the first to study influence maximization as an algorithmic problem. They proposed a probabilistic model of interaction and heuristics were given for choosing individuals with a large overall effect on the network. Kempe et al. [4] studied this problem from discrete optimization perspective and proposed three cascade models for influence propagation. They proposed a greedy algorithm which aimed to find a limited number of influentials from whom the information diffusion can be maximized.

Recently, Leskovec et al. [93, 24] viewed online social networks as *signed* networks, where social interactions involve both positive and negative relationships (edges). For instance, consider the signed network in Figure 4.1 depicting interactions between a set of individuals. An edge pointing from u to v denotes that person u trust/agree (*resp.,* distrust/disagree) person v. An edge representing agreement or trust relationship is labeled as positive (*e.g.,* edge $\overrightarrow{uv}$) whereas the one representing disagreement or distrust is labeled as negative (*e.g.,* edge $\overrightarrow{wv}$). Note that social influence flows in the opposite direction of the edges (*i.e.,* v influenced u and w).

The representation of social interactions as positive and negative relationships demanded a revisit of the social influence analysis problem as general influence analysis techniques overwhelmingly assumed only positive edges among individuals. However, negative relationships between individuals are valuable in several real-world scenarios (*i.e.,* politics) as they often carry as much information as the positive ones. For instance, in 2006, 31% of US residents used the Internet for gathering or sharing political information (above 60 millions people). 28% of them mentioned that most sites they use share their point of views while 29% of the Internet users mentioned that most challenge their point of views [94]. Thus, it is imperative to reconsider social influence analysis problem by taking into account the signed interactions between individuals.

The notion of *conformity* originated in social psychology in the context of social networks. It is defined as yielding to perceived group pressure by copying the behavior and beliefs of others [2]. Several lines of work in social psychology have focused on conducting experiments on groups of people in order to find the cause of conformity from the aspect of human nature [95, 96, 97]. These efforts identified two major causes of conformity, namely, *influential* and *normative*. The former claims that people conform to others' opinions for advice [96]; the latter claims that people conform to others' opinions in order to be consistent with the social group regardless whether the opinion is right or not [97]. In this chapter, we are inspired by the conformity study in social psychology and utilize it to enhance social influence analysis in online social networks. We take into account the conformity in social influence and propose a model to evaluate the conformity of each individual in a social group. However, classification of the causes of conformity within online social networks is beyond the scope of this chapter.

4.1.1 Motivation

A closer analysis of social influence phenomenon in signed networks reveals that there are three important factors that play a key role. Firstly, an individual's ability to influence others (*e.g., v* in Figure 4.1). Secondly, the nature of social interactions (positive or negative) between individuals (*e.g.,* $\overrightarrow{uv}$ and $\overrightarrow{wv}$). Lastly, the degree of *conformity* of an individual, which is a person's inclination to be influenced [2]. The last factor is important as empirical studies have shown that large conversation are not only driven by the limited number of influentials but also by the large population of the early adopters who *accept* the influentials' opinion [98]. For instance, existing work can only conclude that v in Figure 4.1 may be influential in general and thus can influence others no matter whom is the object. However, persons u and w may exhibit different persona in accepting others' opinions which can be summarized as *conformity* [2]. Ignoring this factor may result in an inaccurate conclusion that both u and w will be influenced by v which deviates from the ground truth that u conforms to v while w does not. Moreover, we observe that an individual's ability to influence or conform is *context-sensitive*. For example, reconsider Figure 4.1, where u conforms to v's opinion on `iPad 2` . However, it does not necessarily mean that u will always conform to v on *any* topic. For instance, u may not agree with v on conversation related to `salsa dancing` as u may believe that she is a better dancer than v. Additionally, v may not even be considered as influential on this topic.

Despite the benefits of the state-of-the-art social influence analysis techniques, a key limitation is their inability to systematically exploit the aforementioned second and third factors for superior analysis. Majority of existing research have overwhelmingly focused on considering only the positive relationships. Only very recently Cai et al. [56] took a step towards classifying influential individuals into *Positive Persona*, *Negative Persona*, and *Controversy Persona*, by exploiting both positive and negative relationships in the network. However, they ignored the effect of conformity of individuals on influentials. To elaborate further, consider the two signed networks in Figure 4.2 where an edge $\overrightarrow{uv}$ with positive (*resp.,* negative) sign indicates u trust (*resp.,* distrust) v. The shadowed part depicts the conformity of individuals u_1 and u_2. Specifically, u_1 is easily convinced by others whereas u_2 is not. Thus, it is easier for v_1 to influence u_1 than v_2 to influence u_2. However, state-of-the-art approaches have ignored this issue and failed to differentiate between these two cases. Consequently, these conformity-unaware techniques compute the same *influence score* for v_1 and v_2. However, an individual's influence should be increased if a larger number of users conform to her (positive interactions) but decreased if she is distrusted by individuals who are not conforming to her (negative interactions). *Is it possible to design a conformity-aware social influence analysis strategy that can address the aforementioned limitation?* In this chapter, we provide an affirmative answer to this question.

In a different direction, there are also large bodies of work involving positive (friendly) and negative (antagonistic) links in social media. In [93], the authors analyzed two structural properties of positive edges and negative edges, referred to as *balance* and *status*, in three real-world networks (*Slashdot*, *Epinions* and *Wiki*[1]). *Balance* is a classical theory from social psychology, which postulates that when considering the relationships between three people, either only one or all three of the relations should be positive. *Status* is a theory of directed signed networks which postulates that when person *A* makes a positive link to person *B*, then *A* is asserting that *B* has higher status – with a negative link from *A* analogously implying that *A* believes *B* has lower status. The authors use these two theories to explain the observed edge signs in undirected and directed social networks.

Leskovec et al. [24] used logistic regression to predict the signs of edges in signed networks by exploiting 7-dimensional degree features and 16-dimensional triad features of the networks. They showed that their logistic regression-based method yields higher accuracy in predicting the signs of edges compared to the state-of-the-art. Besides, the authors demonstrated that the accuracy of predicting only the positive edges can be improved by taking into account the negative interactions. In other words, it is often important to consider the interplay between positive and negative interactions. Recently Cai et al. [56] proposed another feature (*i.e.,* influence) aside from the 7-dimensional degree features in [24]. A PageRank-like algorithm was developed to compute the influence of individual users and then use it as another feature in an SVM classifier to predict the signs of edges. They showed that by taking into account the social influence of individual users the accuracy of edge sign prediction can be significantly

[1] `http://www.wikipedia.org/`

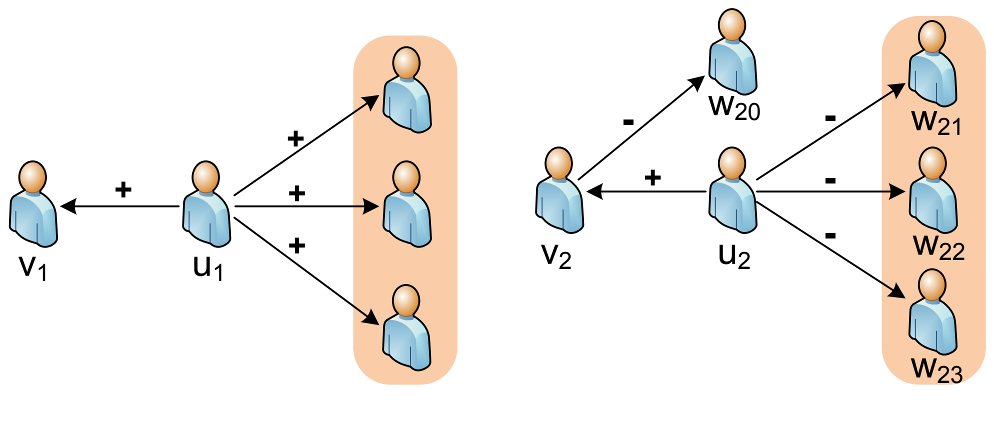


Figure 4.2: Conformity and negative edge effect.

improved. Based on this, they categorized influence personae into *Positive Persona*, *Negative Persona*, and *Controversy Persona*. *Positive* and *Negative Personae* represent users with high positive and negative influence, respectively. The last kind of *Controversy Persona* represents a group of individuals who are liable to be challenged or supported by many.

In the aforementioned research, different structural features of signed networks (*i.e.,* positive/negative edges, in/out degree, influence, triad frequency) are used in machine learning methods in order to predict the signs of edges. However, these research do not address the following two issues. Firstly, although [56] has taken into account the influence of node u for prediction of the sign of edge $\overrightarrow{uv}$, it does not consider the conformity of v. That is, whether v accepts the opinion of u. In contrast, we not only study the influence of u on the sign of edge $\overrightarrow{uv}$ but also investigate the conformity of v and its effect on u's influence. Secondly, an individual v may agree with u on a certain topic as u maybe more knowledgeable than her in that topic. However, v may not agree with u in some other topic where u may not be as proficient as v. In this chapter, we propose a *context-aware* strategy which allows the same individual to exhibit different influence and conformity in different topics.

4.1.2 Overview and Contributions

We propose a novel algorithm called CASINO (**C**onformity-**A**ware **S**ocial **IN**fluence c**O**mputation) that somewhat departs from existing influence analysis techniques in the following way: where existing strategies essentially ignore conformity of individuals, CASINO focuses on integrating the interplay of influence and conformity of individuals for social influence analysis by exploiting the positive and negative signs of edges. Additionally, it is *context-aware*, allowing the same individual to exhibit different influence and conformity over different topics of social interactions.

Given a social network, if it is a *context-aware* one, then CASINO first extracts a set of *topic-based subgraphs*. Each subgraph depicts the social interactions between individuals associated with a specific topic. Since the edges of a social network may not be always explicitly labeled with positive or negative signs, CASINO exploits an existing sentiment analysis technique to label the edges in each topic-based subgraph. Finally, given a set of signed topic-based subgraphs, the algorithm iteratively computes the *influence* and *conformity indices* of each individual in each subgraph.

To validate the proposed algorithm, a series of experiments have been conducted on five public datasets from three popular online social media sites, *i.e., Slashdot*[2], *Epinions*[3], and *Twitter*. *Epinions* and *Slashdot* are *context-free* networks where individuals trust (distrust) each other regardless of any specific topic. The sign of each edge in these networks is explicitly provided. *Twitter* is a *context-aware* network where each conversation is based on a specific topic. Note that the signs of edges are not explicit in *Twitter* and hence they need to be extracted using a sentiment analysis technique.

There is no direct mechanism to justify the list of influentials generated by a generic social influence analysis technique. Hence, to quantitatively evaluate the performance of CASINO, we

[2]http://slashdot.org/

[3]http://www.epinions.com

borrow the experimental framework articulated by Leskovec et al. [24, 93]. In this framework, machine-learning based approach is used for discovering the presence of unknown edges and edge sign prediction, wherein the conformity and influential indices of individuals are used as additional features to describe the signed edges. The intuition for using this framework is as follows. When an individual v influences u, it results in an edge $\overrightarrow{uv}$ in a social network graph. Moreover, if v has more influence on u (u is a conformer), then there is a greater probability of the existence of $\overrightarrow{uv}$. Thus, presence of a positive edge in a social network can be interpreted as the result of influence of v and conformity of u. In summary, the main contributions in this chapter are as follows.

- In Section 4.2, we discuss the roles of influentials and conformers in the context of social influence analysis and describe how to quantify influence and conformity of each individual in a social network. *To the best of our knowledge, we are the first to study the interplay of influentials and conformers with the goal of social influence analysis.*
- In Section 4.3, we propose an iterative algorithm called CASINO that utilizes signs of social interactions (edges) to compute the influence and conformity indices of each vertex in an arbitrary social network. Moreover, we prove that the proposed algorithm is guaranteed to converge.
- By applying CASINO to real-world online social media sites, in Section 4.4, we strongly demonstrate the effectiveness and superiority of CASINO compared to state-of-the-art approaches and at the same time reveal several interesting characteristics of influentials and conformers in these sites.

4.2 Influence and Conformity

In this section, we formally introduce the notion of *influence* and *conformity* in the context of signed social networks. We begin by briefly introducing signed social networks, which lie at

Table 4.1: Symbols.

Symbol	*Semantics*
G	social network graph
V	vertex set
E	edge set
E^+	positive edge set
E^-	negative edge set
A	topic ID
E_A	edge correlated with topic A
G_A	subgraph correlated with topic A
$\mathcal{G}$	topic-based subgraph set
$\Omega(\cdot)$	conformity index
$\Phi(\cdot)$	influence index
$\Omega_A(\cdot)$	conformity index with respect to topic A
$\Phi_A(\cdot)$	influence index with respect to topic A
$\overrightarrow{uv}$	the edge pointing from u to v
$\overrightarrow{uAv}$	the edge pointing from u to v on context topic A

the foundation of our proposed strategy. In the sequel, we shall use the notations shown in Table 4.1 to represent different concepts.

4.2.1 Signed Social Networks

Social interactions in online social networks can be either positive (indicating relations such as friendship) or negative (indicating relations such as distrust and opposition). For instance, in online rating sites such as *Epinions*, people can give both positive and negative ratings not only to items but also to other users. In online discussion sites such as *Slashdot*, users can tag other users as “friends” (positive) and “foes” (negative). In blogosphere and *Twitter*, the reply relationship among users can be a positive or a negative one. In our following discussion, we treat such social interaction as signed directed graph.

In a *signed* social network $G(V,E)$, each edge has a positive or negative sign depending on whether it expresses a positive or negative attitude from the generator of the edge to the recipient [24]. Specifically in this chapter, a positive sign indicates that the recipient supports the opinion of the generator whereas the negative sign represents otherwise. For example, Figure 4.2(b) depicts a signed social network. The positive edge $E^+ = \{\overrightarrow{u_2v_2}\}$ represents trust relationship while the negative ones ($E^- = \{\overrightarrow{w_{20}v_2}, \overrightarrow{u_2w_{21}}, \overrightarrow{u_2w_{22}}, \overrightarrow{u_2w_{23}}\}$) represent distrust relationships. Note that the signs on the edges are not always available explicitly. In networks such as *Epinions* and *Slashdot*, the sign of each edge is explicitly provided. However, in other networks such as blogosphere and *Twitter* the sign of each edge is not explicitly available. In this case, we need to preprocess the network using text mining methods to discover signs associated with the links (detailed in Section 4.3.2). Consequently, a social network $G(V,E)$ containing both positive and negative edges can be represented using a pair of graphs $G^+(V,E^+)$ and $G^-(V,E^-)$ such that the following hold.

$$\forall \overrightarrow{uv} \in E, \begin{cases} (\overrightarrow{uv} \in E^+) \cap (\overrightarrow{uv} \in E^-) = 0, \\ (\overrightarrow{uv} \in E^+) \cup (\overrightarrow{uv} \in E^-) = 1 \end{cases}$$

In other words, $G^+(V,E^+)$ denotes the induced graph of positive edges E^+ (trust/agreement relationship) and $G^-(V,E^-)$ denotes that of negative edges E^- (distrust/disagreement relationship).

4.2.2 Definitions

In our approach, each individual (vertex) in a signed network is associated with a pair of *influence index* and *conformity index* to describe the power of influence and conformity of the individual, respectively. Reconsider the signed network in Figure 4.2(b). Intuitively, a person is influential if there are many people conforming to him. Thus in the scenario, the influence of v_2 should increase when the accumulated conformity of those who trust v_2 (*i.e.,* u_2) increases. On the other hand, a person which is distrust by many people, who always conform to others, should exhibit little influence. Thus, the influence of v_2 should decrease when the accumulated conformity of those who distrust v_2 (*i.e.,* u_2) increases. Hence, the *influence index* of an individual should capture this interplay of influence and conformity and penalize her whenever necessary.

Definition 4.11 *[Influence Index] Let $G^+(V,E^+)$ and $G^-(V,E^-)$ be the induced graphs of the signed social network $G(V,E)$. The influence index of vertex $v \in V$, denoted as $\Phi(v)$, is defined as follows.*

$$\Phi(v) \quad = \quad \sum_{\overrightarrow{uv} \in E^+} \Omega(u) - \sum_{\overrightarrow{uv} \in E^-} \Omega(u)$$

where $\Omega(u)$ represents the conformity index of vertex $u \in V$. ∎

Similarly, the *conformity index* of u_2 in Figure 4.2(b) depends on the influences of vertices which are trusted or distrusted by u_2. Intuitively, as the aggregated influence of those vertices which u_2 trust (*e.g.,* v_2) increases, u_2 is more inclined to conform to others. On the other hand, when the aggregated influence of vertices which u_2 distrust (*e.g.,*, w_{21}, w_{22}, w_{23}) increases, u_2 is less inclined to conform to others. This intuition is captured by *conformity index* which is defined as follows.

Definition 4.12 *[Conformity Index] Let $G^+(V,E^+)$ and $G^-(V,E^-)$ be the induced graphs of the signed social network $G(V,E)$. The conformity index of vertex $u \in V$, denoted as $\Omega(u)$, is defined as follows.*

$$\Omega(u) \quad = \quad \sum_{\overrightarrow{uv} \in E^+} \Phi(v) - \sum_{\overrightarrow{uv} \in E^-} \Phi(v)$$

where $\Phi(v)$ is the influence index of vertex $v \in V$. ∎

Thus, according to the above definition the influence index of v_2 in Figure 4.2(b) can be computed as $\Phi(v_2) = \Omega(u_2) - \Omega(w_{20})$. The conformity index of u_2 is computed as $\Omega(u_2) = \Phi(v_2) - \Phi(w_{21}) - \Phi(w_{22}) - \Phi(w_{23})$. Observe that the aforementioned definitions of influence

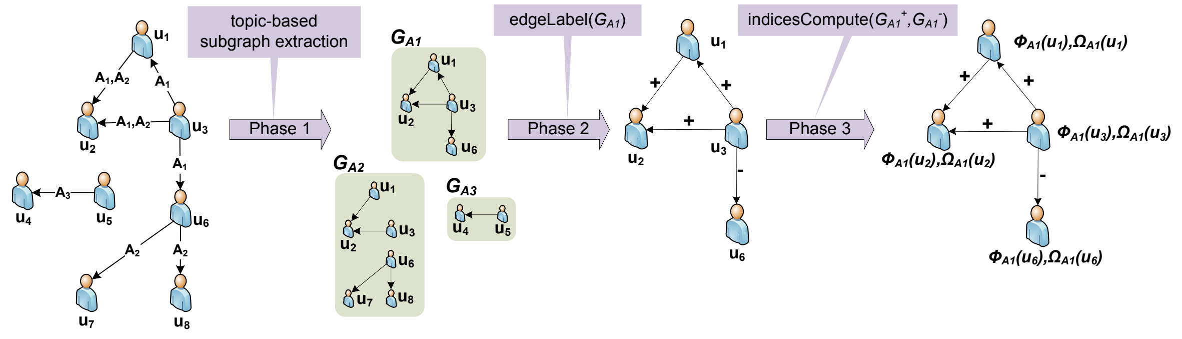

topic-based subgraph extraction
Phase 1
edgeLabel(G_{A1})
Phase 2
indicesCompute(G_{A1}^+,G_{A1}^-)
Phase 3
G_{A1}
G_{A2}
G_{A3}
$\Phi_{A1}(u_1),\Omega_{A1}(u_1)$
$\Phi_{A1}(u_2),\Omega_{A1}(u_2)$
$\Phi_{A1}(u_3),\Omega_{A1}(u_3)$
$\Phi_{A1}(u_6),\Omega_{A1}(u_6)$

Figure 4.3: Overview of CASINO.

and conformity are mutually dependent on each other. Consequently, a recursive computation framework is necessary to compute these two indices. In the next section, we shall present an algorithm called CASINO that computes the influence index and conformity index of every vertex iteratively.

4.3 Conformity-aware Influence Computation

In this section, we formally describe the algorithm CASINO for computing conformity and influence indices of individuals in a social network containing positive and negative edges. We begin by briefly describing the notion of *context-aware* and *context-free* signed social networks to represent real-world online networks.

4.3.1 Context-aware and Context-free Networks

Online social networks can be classified into *context-aware* and *context-free* networks. The former represent networks where the edges are associated with topics (context) as social interactions may often involve conversations on specific topics. For example, each conversation in *Twitter* is based on a specific topic. Figure 4.1 depicts interactions between three users on the topic `iPad 2` . On the other hand, interactions in context-free networks do not involve specific topics. For example, in *Epinions* and *Slashdot* individuals trust (distrust) each other regardless of any specific topic.

Algorithm 3: The CASINO algorithm.

```
Input: Social network $G(V,E)$
Output: the influence index $\mathbb{I}_A = (\Phi_A(u_1), \Phi_A(u_2), \ldots, \Phi_A(u_\ell))$ and conformity index
        $\mathbb{C}_A = (\Omega_A(u_1), \Omega_A(u_2), \ldots, \Omega_A(u_\ell))$ for $V = \{u_1, u_2, \ldots, u_\ell\}$ and for each topic $A$
1 begin
2     if $G$ is context-aware then
3         $\mathcal{G} \leftarrow$ extractSubgraph($G$);
4     else
5         $\mathcal{G} = \{G\}$;
6     foreach $G_A \in \mathcal{G}$ do
7         if $G_A$ is not a signed network then
8             $(G_A^+(V_A, E_A^+), G_A^-(V_A, E_A^-)) \leftarrow$ edgeLabel($G_A$);
9         $(\mathbb{I}_A, \mathbb{C}_A) \leftarrow$ indicesCompute($G_A^+(V_A, E_A^+), G_A^-(V_A, E_A^-)$);
```

The leftmost social network in Figure 4.3 is an example of context-aware social network where an edge labeled as A_1, A_2 indicates that the pair of individuals communicate with each other on topics A_1 and A_2.

4.3.2 The Algorithm CASINO

The CASINO (**C**onformity-**A**ware **S**ocial **IN**fluence c**O**mputation) algorithm is outlined in Algorithm 3 and consists of three phases, namely the *topic-based subgraph extraction* phase (Line 3), the *edge labeling* phase (Line 8), and the *indices computation* phase (Line 9).

Figure 4.3 depicts an overview of the CASINO algorithm. Given a social network $G(V,E)$, if it is a context-aware network then the *topic-based subgraph extraction* phase extracts a set of subgraphs of G (denoted by $\mathcal{G}$) where each subgraph $G_A(V_A, E_A) \in \mathcal{G}$ contains all the vertices and edges in G associated with a specific topic A. Each subgraph G_A represents positive or negative attitudes of individuals toward opinions of others in G with respect to the topic A. In our experiment (*e.g., Twitter*), we can directly acquire a set of posts that belong to a given topic using Twitter API. After that, we construct a sub-graph for each topic by traversing the links between the posts. For instance, in Figure 4.3, this phase generates three topic-based subgraphs, namely, G_{A_1}, G_{A_2}, and G_{A_3}, for topics A_1, A_2, and A_3, respectively. Recall that edges of a social network may not be explicitly labeled with positive or negative signs. This is especially true for context-aware networks (*e.g., Twitter*). On the other hand, links in many context-free networks (*e.g., Slashdot* and *Epinions*) are explicitly labeled with signs. Hence, it is important to label the edges in each topic-based subgraph G_A. The objective of the *edge labeling* phase is to assign sign to each edge by analyzing the sentiment expressed by the generator and recipient of the edge. Figure 4.3 depicts the labeling of G_{A_1}. Finally, given a set of signed topic-based subgraphs $\mathcal{G}$, the goal of the *indices computation* phase is to iteratively compute the influence and conformity indices of each individual in each $G_A \in \mathcal{G}$. Observe that a vertex v in G may have multiple pairs of indices if v is involved in more than one topic-based subgraph. Since the first phase is straightforward, we now elaborate on the remaining two phases in turn.

The edge labeling phase. The edge labeling method varies with dataset. In this chapter, we adopt the method described in Algorithm 4. We denote each edge $\overrightarrow{uv}$ associated with topic A as $\overrightarrow{uAv}$. This enables us to differentiate between an edge which shares the same generator and recipient for more than one topic. For each edge $\overrightarrow{uAv}$ in a topic-based subgraph G_A, we identify 5-leveled sentiment (*i.e.,* like, somewhat like, neutral, somewhat dislike, dislike) expressed at both ends using *LingPipe* [99] which is a popular sentiment mining package adopted in several recent research [100, 101, 26, 102] (Lines 4-5). Note that *LingPipe* has been tested to provide very promising results (*i.e.,* with accuracy over 85% in most cases [26, 100]) on sentiment extraction. If the sentiments at both ends are similar (*sentiment similarity threshold* is less than ε), we denote the edge as positive (Lines 6-7). Otherwise, we denote it as negative (Lines 8-9).

Indices computation phase. Given a topic A and topic-based subgraph G_A, the preceding phase generates G_A^+ and G_A^-. Without loss of generality, assume that there are $|\mathcal{G}|$ different topics. Then, we are able to compute an individual's influence and conformity indices for each topic (*i.e.,* $\Phi_A(u)$ and $\Omega_A(u)$). We now elaborate on the algorithm for computing these indices.

Algorithm 4: The *edgeLabel* procedure.

```
Input: Topic-based subgraph $G_A(V_A, E_A)$ induced by topic $A$,
Output: $G_A^+(V_A, E_A^+)$ and $G_A^-(V_A, E_A^-)$ such that: $E_A^+ \cup E_A^- = E_A$ and $E_A^+ \cap E_A^- = \varnothing$
1 begin
2     $E_A^+ = E_A^- = \varnothing$;
3     foreach $\overrightarrow{uAv} \in E_A$ do
4         $u.sentiment \leftarrow LingPipe.\mathbf{sentExtr}(u)$;
5         $v.sentiment \leftarrow LingPipe.\mathbf{sentExtr}(v)$;
6         if $|u.sentiment - v.sentiment| < \varepsilon$ then
7             $E_A^+ = E_A^+ \cup \{\overrightarrow{uAv}\}$
8         else
9             $E_A^- = E_A^- \cup \{\overrightarrow{uAv}\}$
```

Algorithm 5 outlines the strategy for computing a pair of influence and conformity indices ($\Phi(u)$, $\Omega(u)$) for each vertex u. It first initializes the influence index and conformity index of all vertices to be 1 (Lines 1-4). Subsequently, in each iteration it computes them for each vertex by using the values of the indices in previous iteration (Lines 6-8) and normalizing these values using the square root of the summation of all vertices' index values (Lines 9-13). The algorithm terminates when both indices converge. We shall now prove that the proposed algorithm is guaranteed to converge after a fixed number of iterations n. In other words, the difference between an arbitrary node's indices between n and $n+1$ rounds of iteration is insignificant and hence we do not need to consider additional iterations.

Theorem 4.1 *The indicesCompute procedure described in Algorithm 5 converges.*

Proof: *(Sketch)*

According to Definition 4.11, for each vertex u its influence index $\Phi(u)$ can be computed as the following.

$$\Phi(u) = \sum_{\overrightarrow{u'u} \in E^+} \Omega(u') - \sum_{\overrightarrow{u'u} \in E^-} \Omega(u')$$

Algorithm 5: The *indicesCompute* procedure.

```
    Input: $G(V,E) = G^+(V,E^+) \cup G^-(V,E^-)$
    Output: the influence index $\mathbb{I} = (\Phi(u_1), \Phi(u_2), \dots, \Phi(u_\ell))$ and conformity index
            $\mathbb{C} = (\Omega(u_1), \Omega(u_2), \dots, \Omega(u_\ell))$ for $V = \{u_1, u_2, \dots, u_\ell\}$
1   begin
2       $k = 1$ /*initialize iteration counter*/ ;
3       foreach $u \in V$ do
4           $\Phi^k(u) = \Omega^k(u) = 1$
5       while $\mathbb{I}$ or $\mathbb{C}$ not converged do
6           foreach $u \in V$ do
7               $\Phi_0^{k+1}(u) = \sum_{\overrightarrow{vu} \in E^+} \Omega^k(v) - \sum_{\overrightarrow{vu} \in E^-} \Omega(v)$;
8               $\Omega_0^{k+1}(u) = \sum_{\overrightarrow{uv} \in E^+} \Phi^k(v) - \sum_{\overrightarrow{uv} \in E^-} \Phi(v)$;
9           foreach $u \in V$ do
10              $\Phi^{k+1}(u) = \frac{\Phi_0^{k+1}(u)}{\sqrt{\sum_{v \in V} \Phi_0^{k+1}(v)^2}}$;
11              $\Omega^{k+1}(u) = \frac{\Omega_0^{k+1}(u)}{\sqrt{\sum_{v \in V} \Omega_0^{k+1}(v)^2}}$;
12          $\mathbb{I}^{k+1} = (\Phi^{k+1}(u_1), \Phi^{k+1}(u_2), \dots, \Phi^{k+1}(u_l))$;
13          $\mathbb{C}^{k+1} = (\Omega^{k+1}(u_1), \Omega^{k+1}(u_2), \dots, \Omega^{k+1}(u_l))$;
14          $k = k + 1$;
```

If we denote $\mathbb{I} =(\Phi(u_1), \Phi(u_2), \dots, \Phi(u_\ell))^\top$ and $\mathbb{C} =(\Omega(u_1), \Omega(u_2), \dots, \Omega(u_\ell))^\top$ for $V = \{u_1, u_2, \dots, u_\ell\}$, then the computation of both indices in each iteration can be represented as:

$$\begin{cases} \mathbb{I} = \mathbb{A}_+^\top \mathbb{C} - \mathbb{A}_-^\top \mathbb{C} \\ \mathbb{C} = \mathbb{A}_+ \mathbb{I} - \mathbb{A}_- \mathbb{I} \end{cases}$$

where $\mathbb{A}_+$ and $\mathbb{A}_-$ represent the adjacency matrices for G^+ and G^-, respectively. If we substitute $\mathbb{C}$ in the first equation using the second equation, then the first line turns into the following:

$$\begin{aligned} \mathbb{I}_{k+1} &= \frac{1}{Z}(\mathbb{A}_+^\top - \mathbb{A}_-^\top)(\mathbb{A}_+ - \mathbb{A}_-)\mathbb{I}_k \\ &= \frac{1}{Z}(\mathbb{A}_+ - \mathbb{A}_-)^\top(\mathbb{A}_+ - \mathbb{A}_-)\mathbb{I}_k \end{aligned}$$

Table 4.2: Statistics of context-free datasets.

Dataset	*#nodes*	*#edges*	*#positive edges*	*#negative edges*
Slashdot1	77,357	516,575	396,378	120,197
Slashdot2	81,871	545,671	422,349	123,322
Slashdot3	82,144	549,202	425,072	124,130
Epinions	131,828	841,372	717,667	123,705

Table 4.3: Statistics of the context-aware dataset.

Dataset	*#tweets*	*#trends*	*#tweeters*	*#edges*
Twitter	1,054,261	21,917	576,894	1,230,748

where Z is a normalizing factor such that $\|\mathbb{I}_{k+1}\| = 1$. If we compute $\mathbb{I}_{k+1}$ using $\mathbb{I}_k$ for $k = 1, 2, \ldots, n$ recursively, then $\mathbb{I}_{n+1}$ should be the unit vector along the direction of

$$((\mathbb{A}_+ - \mathbb{A}_-)^\top (\mathbb{A}_+ - \mathbb{A}_-))^n (\mathbb{A}_+ - \mathbb{A}_-)^\top (1, 1, \ldots, 1)^\top.$$

Similarly, $\mathbb{C}_{n+1}$ should be the unit vector along the direction of

$$((\mathbb{A}_+ - \mathbb{A}_-)(\mathbb{A}_+ - \mathbb{A}_-)^\top)^{n+1} (1, 1, \ldots, 1)^\top.$$

According to the result in [103], if M is a symmetric matrix, and v is a vector not orthogonal to the principal eigenvector $\omega_1(M)$, then the unit vector in the direction of $M^k v$ converges to $\omega_1(M)$ as k increases.

Comparing with our case, (1, 1, …, 1)$^\top$ is not orthogonal to $\omega_1((\mathbb{A}_+ - \mathbb{A}_-)(\mathbb{A}_+ - \mathbb{A}_-)^\top)$, thus $\mathbb{C}_k$ converges. Similarly, $\mathbb{I}_k$ also converges.

In summary, both $\Phi(u)$ and $\Omega(u)$ converge. ∎

Observe that the aforementioned technique can easily be extended to compute the aggregated indices of an individual by taking into account the entire social network G over all topics $A = 1, \ldots, |\mathcal{G}|$. In this case, E^+ and E^- in Definitions 4.11 and 4.12 are replaced by $\bigcup_{A=1}^{|\mathcal{G}|} E_A^+$ and $\bigcup_{A=1}^{|\mathcal{G}|} E_A^-$, respectively.

4.4 Experimental Study

In this section, we report experimental results to quantitatively measure the performance of CASINO for discovering influential individuals by considering the conformity of others. To this end, we borrow the experimental framework articulated by Leskovec et al. [24, 93], in which machine-learning based approach is used for discovering the presence of unknown edges and edge sign prediction by exploiting various edge features. By considering conformity and influence indices as new features for classification, it is expected that these additional features will provide more concrete evidence for edge prediction compared to state-of-the-art strategies.

4.4.1 Datasets

We consider the following context-free and context-aware networks where each link is explicitly or implicitly labeled as positive or negative.

Context-free network data. In order to compare the performance of CASINO with state-of-the-art efforts on influence evaluation in signed networks [24, 93, 56], we adopt the same datasets that have been used in these work: *Slashdot* and *Epinions* [4]. Recall that these network data are context-free and contain explicit signs of edges to indicate the attitudes of individuals towards one another. Specifically, we obtained three *Slashdot* datasets at different timepoints. The statistics of each dataset is reported in Table 4.2.

Context-aware network data. We use the *Twitter* dataset to investigate the performance on a context-aware network. The dataset was crawled using the *Twitter* API [5] during Dec 2010 to Feb 2011. We extracted top 20 trends keywords at hourly duration and retrieved up to 1500 tweets for each trend. Then we identified the relationships between all the tweeters in the dataset. Table 4.3 reports the statistics associated with this dataset. Note that these statistics are computed after removing non-English tweets (using *Twitter* API). In order to compute

[4]http://snap.stanford.edu/data/
[5]http://dev.twitter.com/doc

accurate influential and conformity indices, we need to have large context-aware interaction graphs. We removed spam trend keywords which contain only meaningless IDs. Thus, we selected top 492 trends that contain more than 1,000 tweets to compute the indices. For each trend (topic), we identified all the tweets associated to it. Then the edges connecting different tweets using '@' tag are extracted and their signs are assigned as positive or negative using Algorithm 4. Additionally, there exists another tag 'RT' in many tweets indicating that a tweet author supports another author's opinion by re-tweeting it. That is, if an author u directly re-tweets another twitter v, then it indicates that u wants to distribute this tweet to her followers. Hence, we assign positive signs to such re-tweet edges.

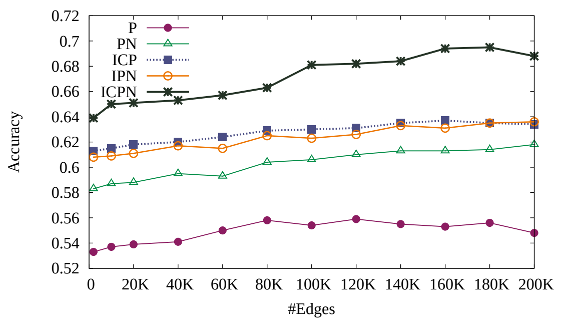
4.4.2 Experimental Setup

We have implemented CASINO using Java and run all the experiments on a 1.86GHz Intel 6300 machine with 4GB RAM with Windows XP. The algorithm converges after average 30 iterations. For the *Twitter* dataset, the sentiments for all the tweets are classified into 5 different levels (*i.e.,* 1-5) using the *LingPipe* library [99]. We set the *sentiment similarity threshold* ε (Algorithm 4) to 1.

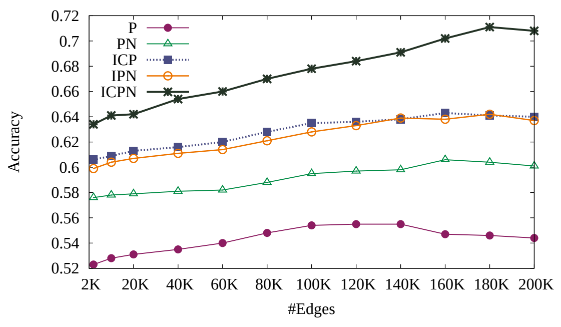
Similar to [24, 93, 56], in our experiments, the task of edge sign prediction is considered as a binary classification problem. We adopted SVM^{light} classifier [104] and classification *accuracy* is taken as the main measure for evaluation. Specifically, accuracy is defined as follows.

$$accuracy = \frac{\#TP + \#TN}{\#TP + \#FP + \#TN + \#FN}$$

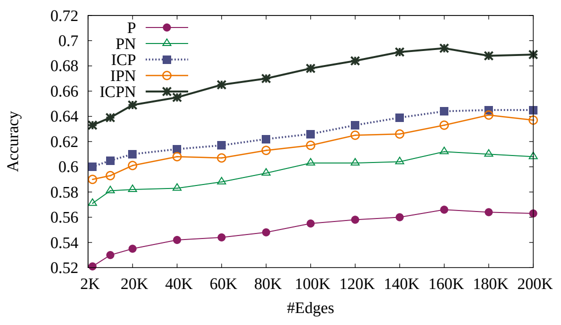
In the above equation TP and TN stand for "true positives" and "true negatives", respectively; FP and FN stand for "false positives" and "false negatives", respectively. The reason for adopting accuracy as evaluation measure over precision and recall is that the former is suitable for quantifying the prediction performances of both positive and negative samples. This is crucial in our framework as in most experiments both positive and negative edges are being predicted.



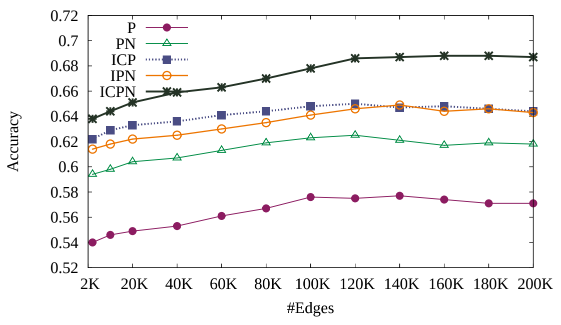
4.4.a: Epinions



4.4.b: Slashdot1



4.4.c: Slashdot2



4.4.d: Slashdot3

Figure 4.4: Positive edge presence prediction.

As mentioned in [24], the overwhelming majority of the edges in *Slashdot* and *Epinions* are positive. Consequently, random guessing can achieve approximately 80% accuracy [56]. In order to avoid such biased classification, we adopt the same strategy used in [24, 93, 56]. Specifically, we create a balanced dataset with same number of positive and negative edges for training and testing.

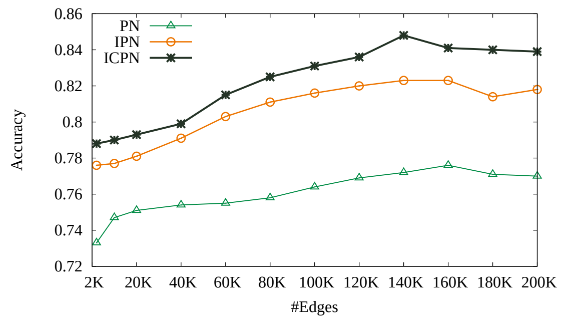
In all experiments we report the average accuracy and perform 5-folds cross validation. For example, for the dataset with 200K edges, we first randomly select 100K negative edges and separate them into 5 parts each of which include 20K negative edges. Then we iteratively select 20K random positive edges and add them into each of the 5 negative edge sets. Based on the learned model (discussed below), we predict a label 1 or -1 for each target edge indicating its

Table 4.4: Features involved in different approaches. (P: positive, N: negative, I: influence, C: conformity, A: topic)

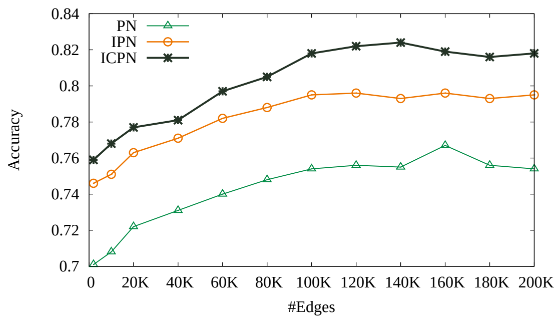
Features \ Approaches	P	PN	IPN	ICP	ICPN	ICAPN
$d_{in}^{+}(v)$	✓	✓	✓	✓	✓	✓
$d_{in}^{-}(v)$	-	✓	✓	-	✓	✓
$d_{out}^{+}(u)$	✓	✓	✓	✓	✓	✓
$d_{out}^{-}(u)$	-	✓	✓	-	✓	✓
$d_{in}(v)$	-	✓	✓	-	✓	✓
$d_{out}(u)$	-	✓	✓	-	✓	✓
$C(u,v)$	✓	✓	✓	✓	✓	✓
$\Phi(u)$	-	-	✓	-	-	-
$\Phi(v)$	-	-	✓	✓	✓	✓
$\Omega(u)$	-	-	-	✓	✓	✓
context A	-	-	-	-	-	✓
$\Phi_A(v)$	-	-	-	-	-	✓
$\Omega_A(u)$	-	-	-	-	-	✓

possibility to be positive or negative, respectively.

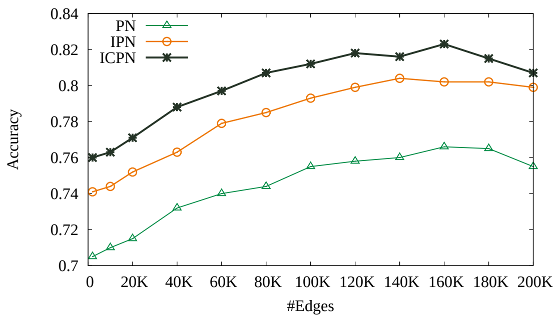
In order to compare the prediction accuracy of the proposed approach with state-of-the-art efforts, a baseline classifier is constructed by referring to the structural features discussed in [24]. Specifically, given an edge from vertex u to v, a 7-dimensional feature vector $\{d_{in}^{+}(v)$, $d_{in}^{-}(v)$, $d_{out}^{+}(u)$, $d_{out}^{-}(u)$, $d_{in}(v)$, $d_{out}(u)$, $C(u,v)\}$ is constructed, where features $d_{in}^{+}(v)$ and $d_{in}^{-}(v)$ denote the number of positive and negative incoming edges to v, features $d_{out}^{+}(u)$ and $d_{out}^{-}(u)$ represent the number of positive and negative outgoing edges from u, features $d_{in}(v)$ and $d_{out}(u)$ denote the number of in-degree and out-degree of v and u, and $C(u,v)$ denotes the total number of common neighbors of u and v without considering the edge direction. Then, we create variants of this baseline classifier by adding influence index and conformity index of vertices as new features in the feature vector. Table 4.4 describes the features involved in the feature vector for each edge $\overrightarrow{uv}$ for different variations of the baseline classifier.



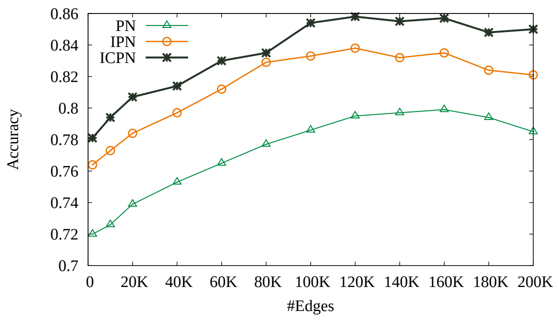
4.5.a: Epinions



4.5.b: Slashdot1



4.5.c: Slashdot2



4.5.d: Slashdot3

Figure 4.5: Signed edge prediction.

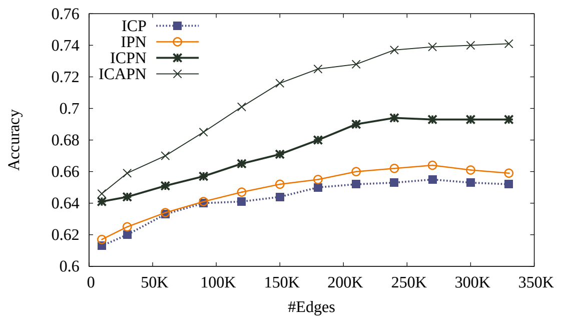
4.4.3 Experimental Results

The goals of the evaluation were to establish whether the proposed approach can reliably predict presence of edges; predict signs of edges; and seek to understand the characteristics and importance of influence and conformity indices for these prediction tasks.

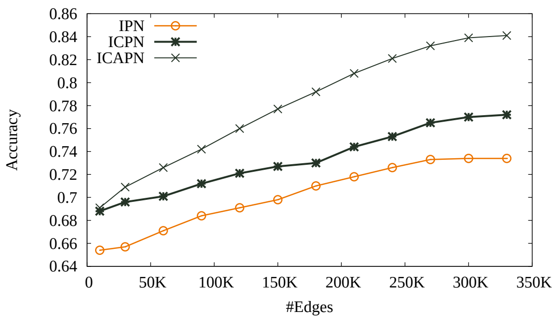
Positive edge presence prediction. We first conduct a series of experiments to predict the presence of positive edges. Note that in order to test the effect of negative edges in predicting the presence of positive edges, we adopted a regression function of SVM^{light} where presence of positive edges are labeled as 1 and negative edges are labeled as -1. We investigate how each of the aforementioned classifiers perform in predicting the presence of positive edges. Figure 4.4 shows the average accuracy for the benchmark datasets, where the results are compared between different classifiers for different size of training and testing data. We can make the following observations. First, for each dataset PN performs better than P indicating that information related to negative edges enhance the quality of edge presence prediction. Second, the prediction accuracy is further improved when we incorporate the influence index or conformity index as a feature (ICP and IPN). Note that ICP performs slightly better than IPN but the improvement is not significant. When the training set is large both approaches exhibit similar performance. Third, ICPN consistently reports the best prediction performance. That is, edge presence prediction is enhanced when we consider the interplay between influentials and conformers.

Signed edge prediction. Next, we undertake a series of experiments to predict the signs of edges. Similar to our earlier experiments, we ensure that the training set and test set both contain equal number of positive and negative edges. We vary the size of training set to test the prediction accuracy. Note that in a binary classification, positive edges and negative edges belong to two different classes. Our goal is to predict the signs of edges which maybe either positive or negative. Figure 4.5 reports the prediction accuracies of the classifiers. Observe that among the three approaches involving both positive and negative edges (PN, IPN, and ICPN), ICPN performs the best, followed by IPN and PN, respectively. Thus, by taking into account the influence and conformity of vertices, the accuracy of sign prediction task can improve significantly.

Edge presence and signed edge prediction in context-aware networks. We now report the performance of our model on a context-aware network (*Twitter*). In this experiment we adopt the classifier ICAPN which takes into account the topic information associated with each edge. That is, the following features for each edge $\overrightarrow{uAv}$ are used to train the model: $d_{in}^{+}(v)$, $d_{in}^{-}(v)$, $d_{out}^{+}(u)$, $d_{out}^{-}(u)$, $d_{in}(v)$, $d_{out}(u)$, $C(u,v)$, $\Phi(v)$, $\Omega(u)$, $\Phi_A(v)$ and $\Omega_A(u)$. Figure 4.6 plots the prediction accuracies of the relevant classifiers. Observe that in both figures ICAPN



4.6.a: Positive edge presence

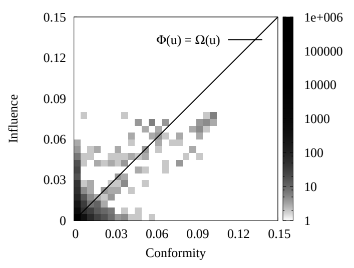


4.6.b: Edge sign prediction

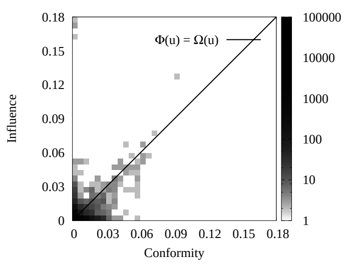
Figure 4.6: Context-aware prediction accuracy.

outperforms the rest. Note that the performances of ICAPN and ICPN are similar when the training set is very small. This is because there may not be enough training edges in each topic-based subgraphs G_A when the training set is very small. Consequently, not enough information is available to accurately compute $\Phi_A(v)$ and $\Omega_A(u)$. All these evidences demonstrate that by leveraging on the influence and conformity indices in topic-based subgraphs, the proposed model leads to superior prediction performance for both positive edge presence and edge sign prediction tasks.

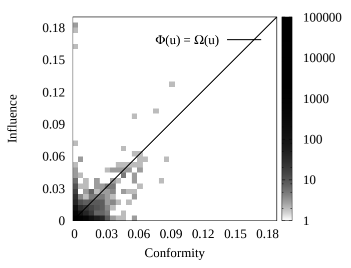
Influentials and conformers. Lastly, we analyze the list of influentials and conformers detected by the CASINO algorithm. Figures 4.7 and 4.8 depict the distribution heatmap of influence index versus conformity index for each benchmark dataset. For each individual u in a network we compute her influence index $\Phi(u)$ and conformity index $\Omega(u)$ and represent it as a point in the influence-conformity 2-D plane. Then we separate the plane into grids of size 0.005×0.005 and count the number of points in each grid. The color shade of a grid denotes the number of points residing in it. Note that both influence index and conformity index are normalized into the range of $[0, 1)$. For each figure, we explicitly draw a boundary line along which the vertices exhibit identical influence index and conformity index. Observe that the line separates the influence-conformity plane into two areas. In the sequel, we refer to the top



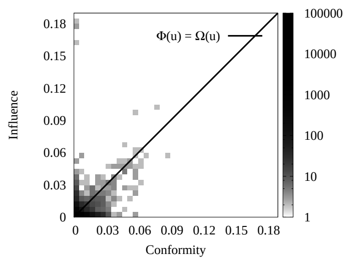
4.7.a: Epinions



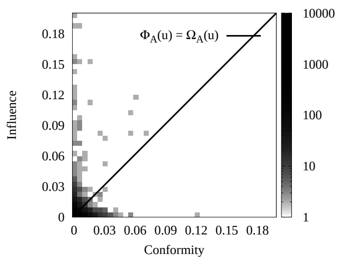
4.7.b: Slashdot1



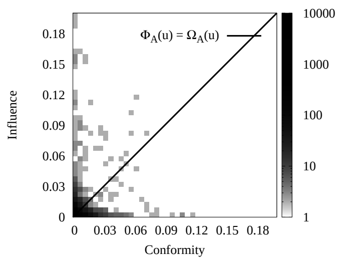
4.7.c: Slashdot2



4.7.d: Slashdot3



4.7.e: Twitter (Top-1)



4.7.f: Twitter (Top-2)

Figure 4.7: Influence vs. conformity distribution heatmap.

area as '*Area I*' and the down one as '*Area II*'. The points belonging to '*Area I*' exhibit higher influence index compared to conformity index, indicating that individuals in this area are more prone to influence others than being influenced. We refer to them as *influence-biased.* On the other hand, the points in '*Area II*' represent individuals who are conforming in nature. That is, they are more prone to be influenced than influencing others. We refer to these individuals as *conformity-biased.*

We first analyze the context-free networks (Figures 4.7(a)-(d)). Consider Figure 4.7(a) related to *Epinions* dataset. Observe that 31% of all individuals belong to '*Area I*'. Consequently, fewer number of individuals in this network are influence-biased. That is, majority of individuals in *Epinions* are often conforming to the others. Similar phenomenon also exists in the *Slashdot* datasets where the percentage of individuals in '*Area I*' is between 36% to 37%.

Observe that those vertices in *Epinions* which exhibit very high conformity index values also have high influence index values. On the other hand, vertices with highest influence index values have a wider range of conformity indices (*i.e.,* from 0 to 0.11). Such phenomenon indicates that in *Epinions* most influence-biased individuals may also conform to others whereas the most conformity-biased individuals are always influencing others. Interestingly, the phenomenon is different in the three *Slashdot* datasets (Figures 4.7(b)- 4.7(d)). Specifically, individuals who are associated with highest influence index values have very small conformity index (*i.e.,* less than 0.02). But individuals with highest conformity index may not exhibit small influence index values. In fact, the conformity index values of these conformity-biased individuals are distributed along the boundary line ($\Phi(u) = \Omega(u)$). Thus, we can make the following observations regarding *Slashdot*. Firstly, there are a few influence-biased individuals who exhibit very high influence but are not easily influenced by others. Secondly, there do not exist conformity-biased individuals who are not influencing others at all.

Next, we analyze the context-aware network (Figures 4.7(e)-(f) and 4.8). Figures 4.7(e)-(f) show the distributions of influence and conformity indices for the top-2 topics (`Mumford & Sons` and `BornThisWayFriday`) with the most number of tweets (4390 and 4046, resp.). Observe that 40% and 45% of all the individuals fall in '*Area I*' for Figure 4.7(e) and (f), respectively. Notably, both these figures exhibit *certain* influence-biased characteristics similar to *Slashdot*. That is, there are a few influence-biased individuals who exhibit very high influence but are not easily influenced by others. This similarity may be due to the fact that both *Slashdot* and *Twitter* are driven by user conversations where majority individuals are commenting or following a few individuals who started the conversations. Figure 4.8 plots the distribution of indices computed over *all* topics. In this case, 41% of all individuals belong to '*Area I*'. The most influential author has influence index of 0.138 whereas the most conforming individual has a conformity index of 0.082.

Table 4.5 shows IDs of top-10 authors who exhibit the highest influence index and conformity index for the top-2 topics as well as for all topics. Consider the top two twitters for all

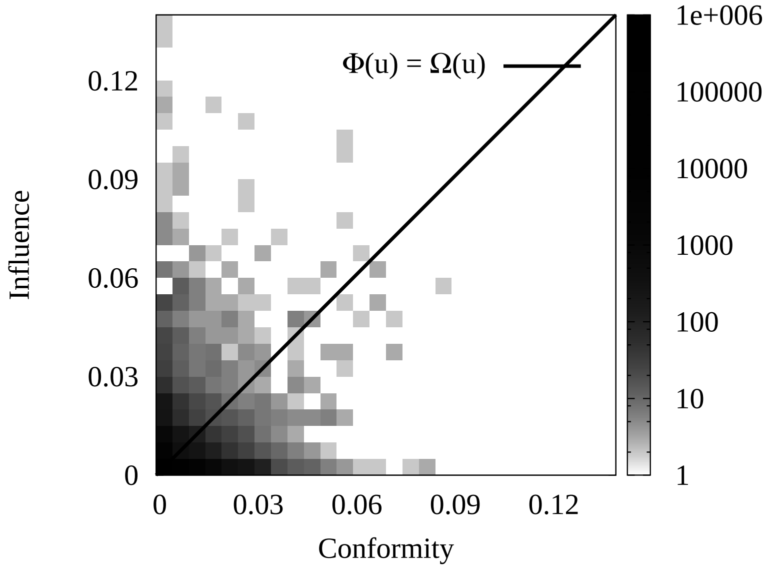


Figure 4.8: Heatmap of *Twitter* for all topics.

topics. The author '142987924' who has the highest influence index receives 66 conforming edges out of 73 in-links over 22 topics. Similarly, the author '49276778' receives 61 conforming edges out of 82 in-links over 24 topics. On the other hand, the author '51389816' who exhibits the highest conformity index initiates 35 conforming edges out of 37 out-links over 37 topics indicating that she has high chance to conform to others' opinions in almost all the topics she is involved in. Furthermore, we can make the following observations. Firstly, none of the top-10 authors occupies a position in *both* indices for each category (*all*, *top-1*, and *top-2*). Secondly, the top-10 individuals having highest influence and conformity indices are different for different topics. This confirms our hypothesis that social influence phenomenon is context-sensitive as same individual may exhibit different influence and conformity over different topics of social interactions.

4.5 Summary

The social influence analysis problem for online social networks, which focuses on studying people's influence by analyzing social interactions between individuals, is considered impor-

Table 4.5: Top 10 authors with the highest indices.

Rank	Influential twitter (#positive in-links/#in-links)			Conformer twitter (#positive out-links/#out-links)		
	All	*Top-1*	*Top-2*	*All*	*Top-1*	*Top-2*
1	142987924 (66/73)	3453454 (13/13)	950596 (31/34)	51389816 (35/37)	121836131 (14/16)	49276778 (101/144)
2	49276778 (61/82)	56068621 (11/11)	190108655 (11/11)	172039151 (31/34)	105332925 (13/14)	202346609 (45/61)
3	119394881 (60/77)	3984874 (10/10)	3498571 (8/9)	177173204 (30/35)	177255919 (11/12)	197538544 (26/30)
4	231134989 (55/71)	133282617 (11/11)	147327886 (5/5)	143062806 (27/34)	193206052 (11/12)	184930795 (22/26)
5	2109823 (56/72)	199855121 (7/7)	49126931 (5/5)	128118710 (25/33)	36525648 (9/10)	148335502 (21/23)
6	92503401 (55/78)	8234375 (5/5)	121158546 (5/5)	130414633 (30/41)	90723076 (7/8)	171387567 (17/20)
7	206661373 (51/66)	1465130 (3/3)	129009252 (5/5)	4782790 (23/30)	123606641 (6/8)	126407259 (18/22)
8	220490093 (46/60)	2894822 (3/3)	79897503 (4/4)	125551983 (22/34)	51513825 (6/6)	114455733 (14/20)
9	168175236 (40/51)	21755211 (2/2)	83629945 (4/4)	91930055 (21/28)	203774695 (5/6)	217826740 (15/20)
10	171287044 (41/62)	4051581 (2/2)	166830172 (4/4)	145339829 (22/31)	203780314 (4/4)	159724683 (12/17)

tant with applications to viral marketing and information dissemination among others. Recently, several techniques have been proposed to address this problem by exploiting the positive interactions (*e.g.,* trust, agreement, friendship) between individuals. However, these techniques ignore two equally important factors that play a key role in social influence propagation, namely, negative relationships (*e.g.,* distrust, disagreement, antagonism) between individuals and conformity of people who are being influenced. In this chapter, we propose a novel algorithm for social influence analysis called CASINO, which quantifies the influence and conformity of each individual in a network by utilizing the positive and negative relationships between individuals.

Our exhaustive experimental study using several online social media sites demonstrates the effectiveness and superior accuracy of CASINO compared to state-of-the-art methods. Specifically, our investigation revealed that the knowledge of conformity of individuals enhance the accuracy of social influence analysis. We also observed several interesting characteristics of influentials and conformers in *Slashdot*, *Epinions*, and *Twitter*. Particularly, in *Slashdot* and *Twitter*, there are a few individuals who exhibit very high influence but are not easily influenced by others. However, in *Epinions*, individuals who exhibit high influence are often conforming to others. Besides, in *Slashdots* and *Twitter* there do not exist individuals who are always influenced but they are not influentials. In contrasts, such individuals can be found in *Epinions*.

Chapter 5

AffRank: Affinity-Driven Ranking of Products

In the previous two chapters, we present our work in the field of individual-level social influence study. In this chapter, we shall discuss our work called *AffRank* (Affinity-Driven Ranking of Products in Online Social Rating Networks) with respect to community-level social influence study. The rest of this paper is organized as follows. Firstly, we briefly introduce the community affinity problem and discussed the limitations of the state-of-the-art technique in Section 5.1. We present a series of novel features that exhibit significant effect in product affinity rank in Section 5.2. We present a prediction model that utilize the proposed features to rank the product communities in Section 5.3. Experimental results are discussed in Section 5.4. Finally, we conclude this chapter in Section 5.5. Preliminary versions of this chapter have been published in [45, 62].

5.1 Introduction

Large online social rating networks (e.g., *Epinions* [1] , *Blippr* [2]) have recently come into being containing information related to many categories of products. Within these websites, individual users are allowed to publish their comments or give ratings on different products. Besides,

[1] `http://www.epinions.com`
[2] `http://www.blippr.com`

they can set up friendships by linking to each other. Figure 5.1 depicts the structure of such a social rating network. Observe that for a particular product (*i.e.,* p_2), there is a group of people who have given ratings and published their comments on it (*i.e.,* u_2, u_3). These people form a community (*i.e.,* c_2). In other words, each product in a social rating network is associated with a community [31]. In the sequel, we refer to such a community as *product community*. Clearly, these communities are a potential gold mine for all kinds of marketing and business analysts as users' comments and ratings toward a particular product may affect other consumers' purchasing behavior [31].

In social rating networks (SRN) a new user can join a product community by giving a *new* rating (review) to the product. Obviously, that particular user should have bought and used the product. To the best of our knowledge, there is no method for us to know exactly whether the user is still using the product or has abandoned it after rating it. On the other hand, the popular way of identifying leaving behavior using an active threshold time t as [87] in general social networks cannot be applied to the examined social rating networks where over 99% of users do not rate a product for a second time. Thus, leaving rate is not explicitly evaluated in these networks. Note that a review that is updated by an existing member is not considered as new. Hence, *the growth of a product community's size can be implicitly measured by the number of new reviews (from new users) it receives during a particular time slot.* We refer to this phenomenon of a product community to "attract" new users by giving new ratings as *product affinity*. Specifically, it is measured by the number of *new* ratings a product receives from *new* members during a particular time period. In fact, existing social rating networks often display a ranked list of products that received the most number of *new* ratings on a daily, weekly, or monthly basis. We refer to this ranked product list at a particular time slot as *affinity rank*. For example, consider Figure 5.2. It depicts two affinity ranks of top-5 movies extracted from the *Blippr* website during weeks $t-2$ and $t-1$, respectively. Observe that *Ninja Assassin* and *Sherlock Holmes* received the most number of new ratings.

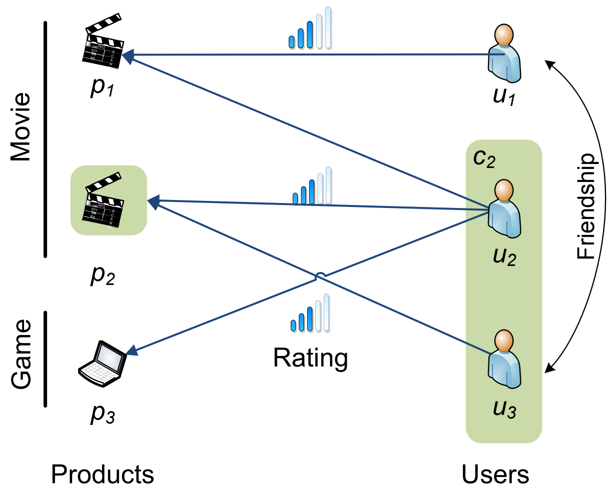


Figure 5.1: Social rating network structure.

It is important to compare the aforementioned notion of product affinity to *affinity* in marketing research, which refers to a marketing strategy [105]. In the latter, *affinity* involves two parties. The first party known as the "affinity group"; seeks to add value to its existing customers, members or donors by promoting products and services they do not currently sell (e.g., financial services). The second party known as the "product supplier"; seeks to acquire new customers by using the strength of another organization's relationship with its customers, through which it aims to distribute its product or service. In other words, the aim of affinity marketing is to build and develop new customer relationships through the existing distribution channels of a third party. In contrast in SRN, a product community is similar to an affinity group as the former promotes (or demotes) a product that they do not sell to its members by giving reviews or ratings. On the other hand, the "product supplier" in SRN seeks to attract new customers by exploiting the *affinity strength* of an "affinity group" (product community). In this chapter, we undertake a quantitative study to analyze and predict the affinity strengths of product communities.

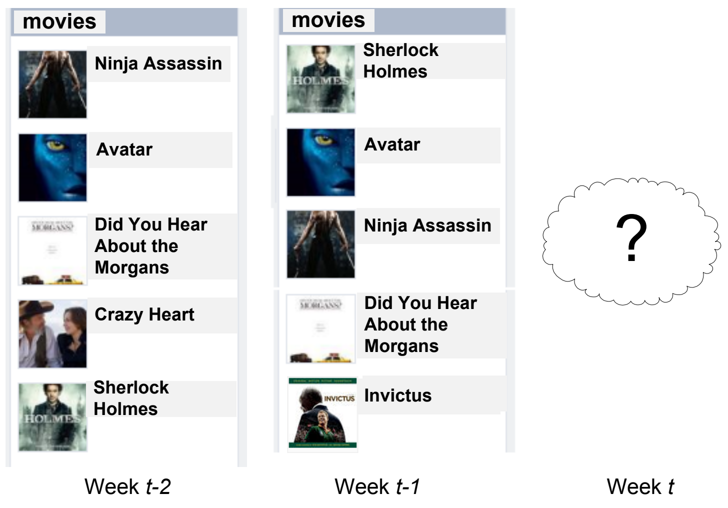


Figure 5.2: Product affinity and affinity ranks.

5.1.1 Motivation

The affinity ranks of products in the past weeks/months highlight the reviewers' affiliation of products in the (recent) past. Although such historical information is important for several applications, prediction of future affinity ranks of products is even more important to marketing and business strategists. For example, reconsider Figure 5.2. Suppose in week $t-1$ a company intends to put advertisements on 5 movie products during week t. Then, it makes sense to predict 5 most popular products at time t so that optimum benefit can be achieved. That is, it is desirable to predict the affinity ranks of products in the near future. In this context, instead of just listing most popular products, ranking them based on their affinity ranks makes more sense as a company may allocate different shares of their advertisement budget depending on the popularity of the products. Note that such top-k products may vary considerably at two different time points. For instance, consider the top-5 products during weeks $t-2$ and $t-1$ in Figure 5.2. Observe that *Sherlock Holmes* moved from rank 5 to the top rank in successive weeks. Further, *Invictus* first appeared in the top-5 list in week $t-1$ whereas *Crazy Heart* failed to remain in the top-5 list in this week.

Recent research on ranking products in SRN have primarily focused on evaluating a product by the strength of connections among its users [31] or by the features related to a particular product item (*i.e.,* price, released time etc.) [106]. However, these techniques are not designed to predict the ranks of products based on their affinities. In this chapter, we propose a novel quantitative model called *AffRank* that utilizes historical and evolutionary affinity information as well as other features to predict the future ranks of products according to their affinities.

5.1.2 Overview

The problem of predicting and ranking product affinity is related to recent efforts [38, 107, 108, 109, 110, 111] in predicting community growth as the former is influenced by the number of new members joining the community. Specifically, these efforts reveal that the community size, connectivity between community members, number of friends a user has in a community [38], and similarity of interests a member has with a community have strong influence on the growth of communities [107]. Hence at a first glance, it may seem that these features should also strongly influence the product affinity. However, our study demonstrates that *the effects of these features are negligible in product community* as it is not strictly similar to traditional social network communities (*e.g.,* users are sparsely connected). Hence we propose three additional features, namely *affinity rank history*, *average ratings*, and *affinity evolution distance*, related to the product communities that may exert significant effect on affinity. In particular, *affinity rank history* represents the historical affinity ranks of products over time; *average ratings* measures the average of ratings received by a product at a given time point; and *affinity evolution distance* measures the distance between affinity evolution of a pair of different products. *To the best of our knowledge, these features have not been studied systematically in the literature.*

In Section 5.4, we evaluate the performance of all the aforementioned features and further compare the three new features with the traditional features for affinity rank prediction. Additionally, our investigation with two real-world datasets (*Epinion* and *Blippr*) revealed several interesting and novel findings related to these three features. For instance, we observe that product affinity is more likely to *increase spikily* and *drop down smoothly*. Further, there is $0 \sim 3$ days lag between the *peak time* of affinity and users' ratings. Also, average *distances* between affinity evolution of products in the *same* category are always smaller than those from *different* categories.

Based on these findings, we propose the *AffRank* model to predict the future affinity rank of a product. Experiments conducted over real-world datasets demonstrate that our prediction and ranking scheme generates high quality results and outperforms several baseline methods. In summary, the main contributions in this chapter are as follows.

- In Section 5.2, we investigate an array of features that exert effect on the evolution of product affinity and affinity rank prediction. To the best of our knowledge, we are the first to study the evolution of product affinities with the goal of predicting affinity ranks.
- In Section 5.3, we formulate the task of ranking product affinity as an autoregressive problem with exogenous input features. We propose a quantitative model called *AffRank* that utilizes historical ranks and evolutionary product affinity information to predict the future ranks based on product affinity.
- By applying *AffRank* to real-world datasets, in Section 5.4, we show its effectiveness and superiority of its prediction quality compared to baseline methods. Further, our experimental results show that traditional community features (*e.g.,* community size, member connectivity, and social context) presented in [38, 107, 108, 109] have negligible influence on product affinities. Instead, features such as affinity rank history, affinity evolution distance, and average rating exert significant influence on affinity rank prediction.

5.2 Features for Affinity Prediction

In this section we describe the features that are used in our ranking model. We begin by introducing the real-world datasets we have used for our study. Then for the sake of completeness,

Table 5.1: Symbols and Semantics.

Symbol	*Semantics*
$u_1, \ldots, u_n \in \mathbb{U}$	users
$p_1, \ldots, p_m \in \mathbb{P}$	products
$c_1, \ldots, c_m \in \mathbb{C}$	communities
$\mathbf{C}_i^t$	users in c_i at time slot t
$\mathbf{F}_j$	friends of user u_j
$\mathbf{S}_i^t$	friends set of $\mathbf{C}_i^t$
$\mathbf{U}_i^t$	set of new users on product p_i at time slot t
$\mathbf{R}_i^t$	bag of new ratings on product p_i at time slot t
$\overline{\mathbf{R}}_i^t$	average rating
a_i^t	number of new users for p_i at time slot t
$\alpha_i(t)$	affinity intensity of p_i at time slot t
$\rho_t(p_i)$	affinity rank of product p_i at time slot t
$r_t(p_i)$	predicted value of $\rho_t(p_i)$

Table 5.2: Statistics of the datasets.

Dataset	*#products*	*#categories*	*#ratings*	*#users*	*#user-user links*
Blippr	75	5	8,032	2,219	10,480
Epinions	678,725	12	13,362,381	132,000	242,831

we briefly describe existing community-based features proposed in the literature that influence the community size. Next, we propose new features for addressing the affinity rank prediction problem. All the values for these features are normalized into the interval [0,1] using Min-Max Normalization [112]. In the sequel, we shall use the notations shown in Table 5.1 to represent different concepts.

Table 5.2 describes two real-world datasets that are used in this chapter. The *Blippr* dataset was crawled using *Blippr* API [3] till August, 2009. It includes user ratings toward 75 different products. The *Epinions* dataset was downloaded from TrustLet [4]. It contains ratings proposed during 2001. Additionally, we crawled the *Epinions* website to retrieve product category and user-user relationships information as TrustLet dataset does not provide them.

[3] http://api.blippr.com/v2/

[4] http://www.trustlet.org/

5.2.1 Traditional Features

Recall from preceding section, several previous work have demonstrated the existence of correlation between characteristics of a community and its affinity. These characteristics include the community size, connectivity between community members, number of friends a user has in a community [38, 25, 113], and the similarity of interests people have with a community [108, 109]. We refer to these features as traditional features as they are associated with communities in conventional social networks. In this section, we describe in detail how we utilize and compute these traditional features in the product community.

Community size. The relationship between community size and its affinity has been studied in several work [38, 39]. Therefore, we incorporate *community size* in our model to predict the future affinity ranks of products. Formally, the size of a product community at time t is calculated as $|\mathbf{C}_i^t|$.

Member connectivity. According to a recent study [38], people are attracted to a community not only because they have friends in it but also the close connections among friends. Thus, the connectivity within the community affect the community's affinity. Clustering coefficient [38, 25, 114, 115] is widely adopted to measure the connectivity within a community and is computed as follows.

$$CC_i(t) \quad = \quad \frac{|3 \times \text{closed triplets in } \mathbf{C}_i^t|}{|\text{triplets in } \mathbf{C}_i^t|} \qquad \text{(Eq. 5.1)}$$

We compute the clustering coefficient for each community and use them to predict the future affinity rank.

Social context. According to some existing studies [107, 108, 109, 116, 117], people are more probable to join a community he/she is interested in. *Social context* represents the similarity in the interest between the members of the community and their friends who are not in the community yet. The friends of users in $\mathbf{C}_i^t$ can be computed by the following equation.

$$\mathbf{S}_i^t = (\bigcup_{u_j \in \mathbf{C}_i^t} \mathbf{F}_j) \setminus \mathbf{C}_i^t. \qquad \text{(Eq. 5.2)}$$

To measure the similarity in the interests between $\mathbf{C}_i^t$ and $\mathbf{S}_i^t$, we first define *interest-similarity* between a pair of users. Let $Int(u)$ be the set of product categories user u is interested in. Then, the *interest-similarity* between users u and v is defined as follows.

$$sim(u,v) = \frac{|Int(u) \cap Int(v)|}{|Int(u) \cup Int(v)|} \qquad \text{(Eq. 5.3)}$$

Using the above similarity measure we can compute the interest-similarity between two groups of users $\mathbf{S}_i^t$ and $\mathbf{C}_i^t$. The *interest-similarity* between two sets of users U and V is defined as follows.

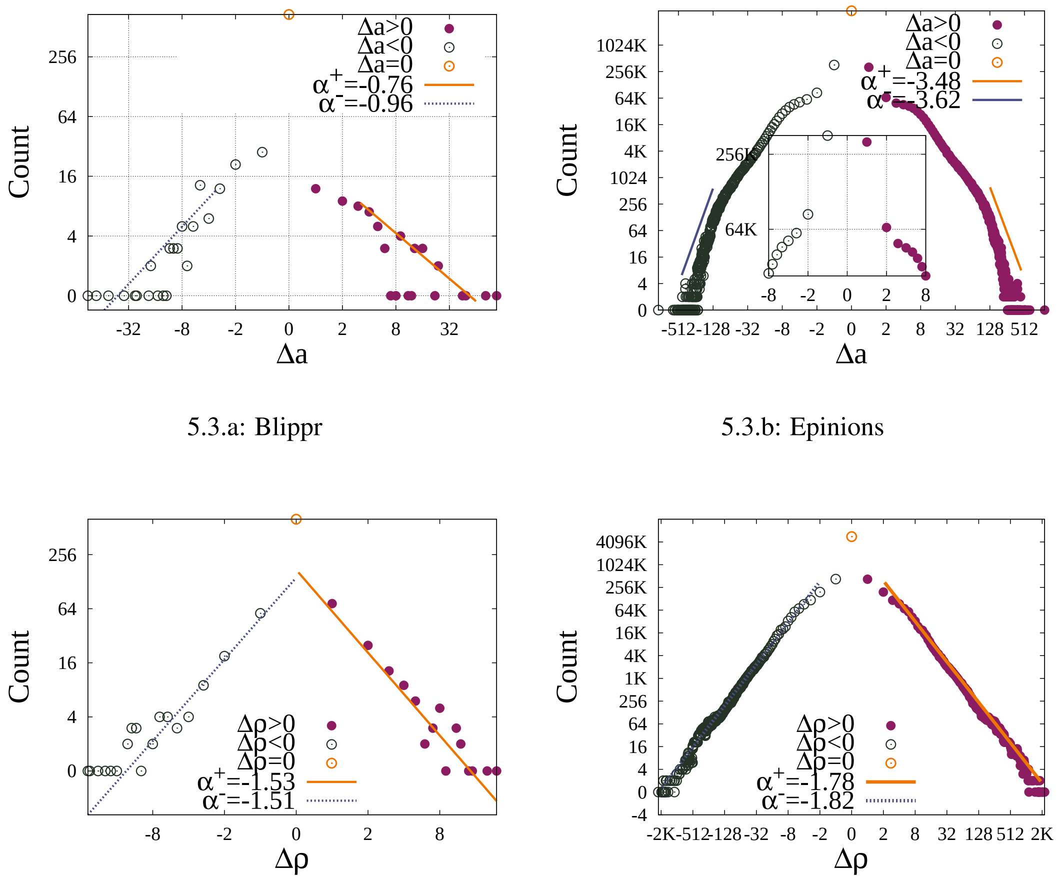
$$simC(U,V) = \sum_{u \in U, v \in V} sim(u,v) \qquad \text{(Eq. 5.4)}$$

In our model, we calculate $simC(\mathbf{C}_i^t, \mathbf{S}_i^t)$ for each product community at time t and then normalize it to [0,1] as of other features.

5.2.2 Affinity Rank History

We now present three new features related to product communities that have been ignored in the literature, namely *affinity rank history*, *affinity evolution distance*, and *average rating*. We begin with the *affinity rank history*.

Recall that (Section 5.1) each product p_i is associated with an affinity rank at time t, denoted by $\rho_t(p_i)$. For example, in Figure 5.2 the affinity ranks of `Sherlock Holmes` movie is 5 and 1 in weeks $t-2$ and $t-1$, respectively. Since the affinity rank of a product depends on the number of new users (product affinity), here we characterize the evolutionary behaviors of product affinity and affinity rank. We first investigate how the affinity changes between consecutive time points in the history. We denote the number of new users for product p_i at time slot t as



5.3.a: Blippr

5.3.b: Epinions

5.3.c: Blippr

5.3.d: Epinions

Figure 5.3: Distributions of Δa and $\Delta\rho$: $\Delta\rho$ show symmetrical patterns while Δa do not.

$a_i^t = |\mathbf{C}_i^t| - |\mathbf{C}_i^{t-1}|$. Figures 5.3(a) and 5.3(b) report the distribution of $\Delta a_i^t = a_i^t - a_i^{t-1}$ over all products p_i and time t. Observe that in Figure 5.3(a) the count of cases where $\Delta a \in [-1,-2]$ is more than that of $\Delta a \in [1,2]$. Besides, the absolute value of power law exponent for the tail at negative side ($\alpha^- = -0.96$) is bigger than that of the positive side ($\alpha^+ = -0.76$). It indicates that the positive affinity change is more likely to have bigger Δa than the negative one. We observe the same phenomenon for *Epinions* dataset in Figure 5.3(b). Specifically, the count of negative changes is larger than the positive ones when $\Delta a \in [-8,8]$; the absolute value of power law exponent for the tail at negative side ($\alpha^- = -3.62$) is bigger than that of the positive side ($\alpha^+ = -3.48$). Both of the above phenomena indicate that the product affinity for all the products discussed in this chapter is more likely to *increase spikily* and *drop down smoothly*. As affinity is the number of new users associated to a product within a time interval, this phenomenon reflects the speed of growth of product community size. It suggests that the speed of growth tends to increase to a peak in a short time and diminishes slowly subsequently. The aforementioned phenomenon is generally applicable to most products, although we do acknowledge that in some specific categories (e.g., car batteries) this may not be true. However, investigating why such phenomenon occurs and to what extent it is applicable to different products is orthogonal to the problem addressed in this chapter. In the following, we show that the change of affinity rank is symmetric while that of affinity is asymmetric according to above discussion.

We now investigate the change of affinity rank between consecutive time slots in the history for each product. In particular, the change of affinity rank for a product between times $t-1$ and t is measured as follows.

$$\Delta\rho_t(p_i) \quad = \quad \rho_t(p_i) - \rho_{t-1}(p_i). \qquad \text{(Eq. 5.5)}$$

We calculate the count for each $\Delta\rho_t(p_i)$ value over all products and time slots. The distribution of $\Delta\rho$ is reported in Figures 5.3(c) and 5.3(d). Clearly, it follows a long-tail distribution. If we fit the curve in Figure 5.3(c) using power law models at both the left and right sides of $\Delta\rho = 0$ separately, the exponents equal to -1.51 for the negative changes where $\rho_{t-1}(p_i) > \rho_t(p_i)$ and -1.53 for the positive ones. The exponents are so close that the distributions of positive $\Delta\rho$ and negative $\Delta\rho$ are almost symmetrical. Figure 5.3(d) shows that *Epinions* dataset also exhibits the same behavior ($\alpha^+ = -1.78, \alpha^- = -1.82$). Besides, such a phenomenon also indicates that the change of rank is most probable to be within a small range. We observe that the aforementioned phenomenon also exists for each specific category of products in both datasets. Table 5.3 reports this phenomenon for a representative set of product categories. Thus, the rank of a product at any time slot is highly related to that of the previous time slot.

Table 5.3: Exponents in each specific category.

	Blippr					Epinions				
	M&T [5]	*Game*	*Music*	*Book*	*Appln* [6]	*Media*	*Cars*	*Elecs* [7]	*Games*	*H&G* [8]
α^+	-2.03	-1.50	-1.37	-1.30	-1.54	-1.80	-1.82	-1.78	-1.77	-1.78
α^-	-1.92	-1.51	-1.36	-1.35	-1.50	-1.81	-1.81	-1.83	-1.81	-1.82

In summary, the distribution of Δa is asymmetrical over the positive and negative sides while the distribution of $\Delta\rho$ is symmetrical. In the next section, we shall exploit the symmetric property of $\Delta\rho$ instead of asymmetric Δa in our *AffRank* model. As we shall see in Section 5.4, our *AffRank* model outperforms the approach based on Δa.

5.2.3 Affinity Evolution Distance

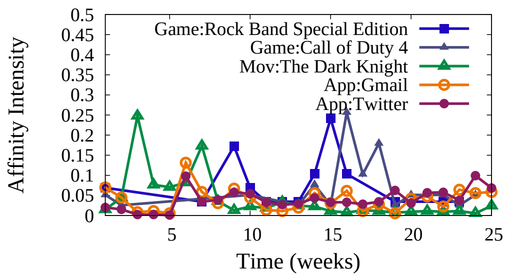
We now analyze and compare the evolutionary nature of different product affinities. We begin by introducing the notion of *affinity intensity*. Recall from Section 5.1, new users join a product community by giving new ratings.

Definition 5.13 ***[Affinity Intensity]*** *Let $|\mathbf{U}_i^t|$ be the number of new users towards product p_i at time slot t. Then the affinity intensity of p_i at t is defined as:*

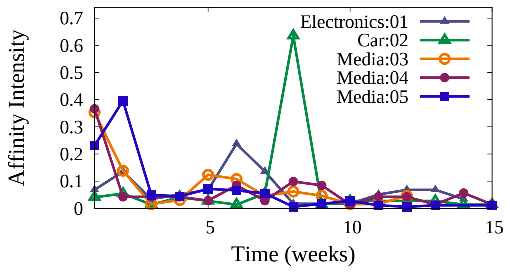
$$\alpha_i(t) = \frac{|\mathbf{U}_i^t|}{\sum_{\tau=1}^{T} |\mathbf{U}_i^\tau|} \qquad \text{(Eq. 5.6)}$$

Note that $\mathbf{U}_i^t$ represents the set of new users towards product p_i at t. In the above definition, T is the number of time slots over which the affinity is normalized. The affinity intensity can be viewed as a normalized histogram where the values in each bin sum up to 1.

[7]*Movie & TV*
[8]*Application*
[9]*Electronics*
[10]*Home & Garden*



5.4.a: Blippr



5.4.b: Epinions

Figure 5.4: Affinity intensity evolution of different products.

Figure 5.4 reports the evolution of affinity intensity values of five different products over time. Note that the label in front of a product name indicates the category of the product. The x-axis denotes the number of weeks since the product first appeared in the website, while y-axis represents the affinity intensity towards the product over different weeks. In the sequel, we refer to such curve as *Affinity Intensity Curve* (AIC). In the *Blippr* website (Figure 5.4(a)), *Gmail* and *Twitter* belong to the same category (`online applications`). *The Dark Knight* belongs to the `movie` category while the other two belong to the `game` category. Observe that out of the five AIC in Figure 5.4(a), the AIC of *Twitter* and *Gmail* look *similar*, the two products of `game` also exhibit similar AIC while that of *The Dark Knight* is quite different from the rest. We observe similar phenomenon for the *Epinions* website as well (Figure 5.4(b)). In other words, *these curves show that products in the same category tend to have similar affinity evolution patterns.* We now quantitatively measure the distance between affinity evolution patterns and investigate if this hypothesis holds.

As the AIC is a one-dimensional time series data, we can compute the distance between different curves using Dynamic Time Warping (DTW) distance. DTW distance is widely used to match similar time series data. In this chapter, we adopt the DTW distance with Sakoe-Chiba band [118] which adds a window constraints w to the warping path found by DTW algorithm. As a result, the DTW algorithm will only match similar shaped data series that have small

Table 5.4: Avg. DTW distance between different categories (Blippr)

	M&T	*Game*	*Music*	*Book*	*Appln*
M&T	**0.5376**				
Game	0.6235	**0.5896**			
Music	0.6891	0.6884	**0.6565**		
Book	0.7324	0.6519	0.6765	**0.5993**	
Appln	0.7088	0.6348	0.7414	0.6877	**0.4455**

Table 5.5: Avg. DTW distance between different categories (Epinions)

	Media	*Cars*	*Elecs*	*Games*	*H&G*
Media	**0.4268**				
Cars	0.5975	**0.5226**			
Elecs	0.7214	0.6194	**0.5837**		
Games	0.7067	0.6322	0.6215	**0.6071**	
H&G	0.6125	0.6571	0.6732	0.7025	**0.4455**

displacement within window w.

We compute the DTW distance between each pair of AIC. We fix the length of each data sample to be $T = 25$ weeks. We set $w \in \{T/4, T/5, T/6\}$. Interestingly, the average distances between products in the same category are always smaller than those from different categories for all values of w. Table 5.4 shows the average DTW distances between representative product categories in *Blippr* (for $w = T/5$). Similar phenomenon is observed in *Epinions* (Table 5.5).

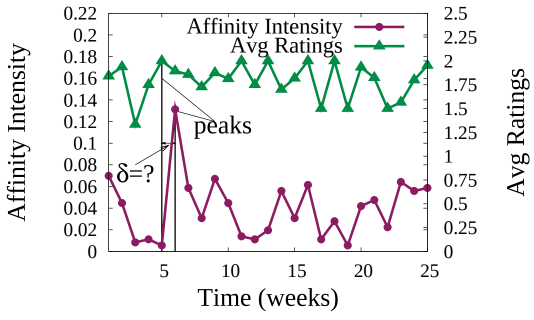
We now elaborate on how we compute *affinity evolution distance* for a product p_i at time t (denoted by ϕ_i^t) using the notion of DTW distance. Since we intend to measure how similar an AIC of a product is compared to the product in the same category with most number of new users, we quantify ϕ_i^t by measuring the DTW distance between the AIC of a product p_i and the AIC of product p_j whose affinity rank is the highest in the same category. The procedure for computing ϕ_i^t is shown in Algorithm 6.

5.2.4 Average Rating

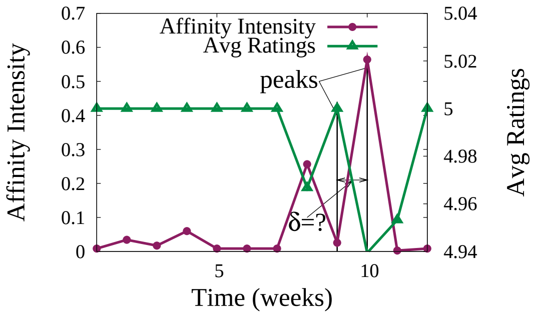
Observe that the affinity evolution distance feature describes evolutionary relationship between different products. We now focus on each individual product and investigate the relationship

Algorithm 6: Affinity evolution distance.

```
  Input: affinity $a_i^j$ for all products $p_i \in \mathbb{P}$ and $j \in [t-T, t-1]$,
  Output: the value of affinity evolution distance: $\phi_i^t$ for products $p_i \in \mathbb{P}$ at time $t$
1 begin
2     forall the $p_i \in \mathbb{P}$ do
3         compute affinity intensity $\alpha_i(j)$ according to Definition 5.13 for $j \in [t-T, t-1]$;
4         $\lambda_i = i$;
5         forall the $p_j \in \mathbb{P}$ and $p_j.category = p_i.category$ do
6             if $\rho_{t-1}(p_j) < \rho_{t-1}(p_{\lambda_i})$ then
7                 $\lambda_i = j$
8     forall the $p_i \in \mathbb{P}$ do
9         $\phi_i^t = DTW([\alpha_i(t-T), \ldots, \alpha_i(t-1)], [\alpha_{\lambda_i}(t-T), \ldots, \alpha_{\lambda_i}(t-1)])$;
```



5.5.a: *Gmail* in Blippr



5.5.b: *Desitin Diaper* in Epinions

Figure 5.5: Lag between the average rating evolution and affinity intensity curve.

between a product's AIC and the evolution of *average ratings* it received. Note that users' ratings towards each product are explicitly provided in both datasets. In the case when the ratings are not explicitly provided in a specific dataset, an existing sentiment analysis model [119, 120] can be used to extract and quantify the users' comments into ratings. Extraction and quantification of such sentiments from users' comments is orthogonal to this work and hence it is not discussed here.

Definition 5.14 ***[Average Rating]*** *Let* $\mathbf{R}_i^t$ *be the bag of ratings that product* p_i *received during*

Table 5.6: Correlation between previous ratings and current affinity intensity (Blippr)

Lag time (in weeks)	*M&T*	*Game*	*Music*	*Book*	*Appln*
0	**0.6328**	**0.6737**	**0.8216**	**0.7347**	**0.7183**
1	0.1691	0.1745	0.1856	0.0860	0.4965
2	0.1132	0.0252	0.0484	0.0601	0.4121
3	0.0542	0.1460	-0.0600	0.0936	0.4319
4	0.0590	0.0608	0.0057	0.0418	0.3379

Table 5.7: Correlation between previous ratings and current affinity intensity (Epinions)

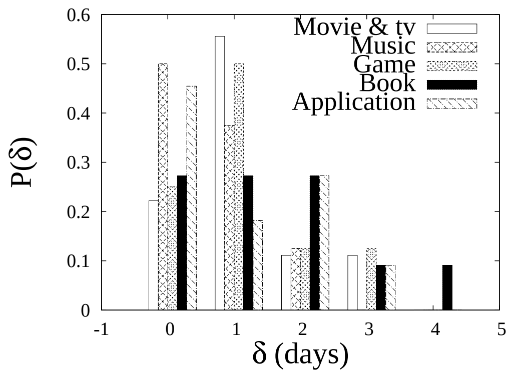
Lag time (in weeks)	*Media*	*Cars*	*Elecs*	*Games*	*H&G*
0	**0.4572**	**0.7101**	**0.7496**	**0.6883**	**0.6015**
1	0.2106	0.1061	0.1374	0.1140	0.1700
2	0.0892	0.0545	0.1007	0.1032	0.2093
3	0.0770	0.0819	0.1220	0.1529	-0.1059
4	-0.0600	-0.0940	0.1019	0.0429	0.1731

time slot t. Then the average rating of p_i during t, denoted by $\overline{\mathbf{R}}_i^t$, is defined as:

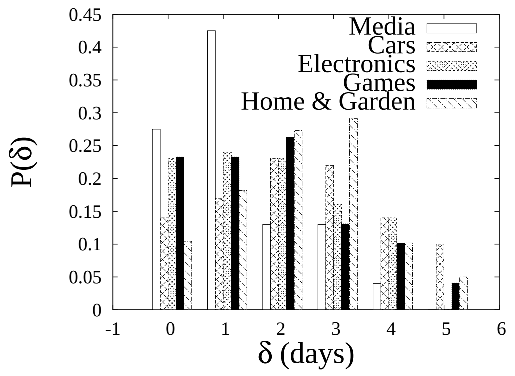
$$\overline{\mathbf{R}}_i^t = \frac{1}{|\mathbf{R}_i^t|} \sum_{r \in \mathbf{R}_i^t} r. \qquad \text{(Eq. 5.7)}$$

Figure 5.5(a) depicts the AIC of *Gmail* in *Blippr* dataset as well as its average rating evolution. Observe that the evolutions of affinity intensity and average rating follow similar trend except that there is a certain delay δ between the peaks of the two curves. Similarly, Figure 5.5(b) shows the same phenomenon in *Epinions* dataset. In the following, we conduct a series of analysis on δ to characterize the relationship between users' ratings and affinity intensity.

Firstly, we study the correlation between the affinity intensity at time t and the average ratings at times $t-\delta$ (weekly). The correlation coefficients using different δ on the *Blippr* dataset are shown in Table 5.6. The affinity intensity is correlated with the average rating in the same week (week 0) with an average correlation coefficient of 0.7221. It is much higher



5.6.a: Blippr



5.6.b: Epinions

Figure 5.6: Probability distribution of δ.

than the correlation with average ratings in any of the previous weeks. The same phenomenon exists in *Epinions* as shown in Table 5.7. Thus, we can conclude that δ is less than a week.

Next, in order to find the exact value of δ, we conduct another set of experiments. For each product, we detect the first peaks [9] in both the AIC and average rating curve by days. After that, we compute the interval between the peaks of these two curves for each product. Finally, we calculate the probability distribution of δ according to the following equation.

$$P(\delta = n) = \frac{\left|\{p_i | Peak_\alpha(p_i) - Peak_{\overline{R}}(p_i) = n\}\right|}{|\{p_i\}|}. \qquad \text{(Eq. 5.8)}$$

where $Peak_\alpha(p_i)$ (resp. $Peak_{\overline{R}}(p_i)$) represents the day when the first peak appears in the AIC (resp. average rating evolution curve) of product p_i. The probability distribution of δ separated by category is reported in Figure 5.6. Observe that in *Blippr* the δ values for `music` are most likely to be 0 while the distribution of δ for `book` is stable across $[0,2]$. Such a phenomenon may be due to the characteristics of products in various categories. Intuitively, the average ratings for musical product can take instant effect on future affinity intensity as potential users can listen to the music (often online) as soon as they see others' ratings. However, potential

[9] We use the Matlab function "*peakdet*" downloaded from `http://www.billauer.co.il/peakdet.html` to detect the peak.

readers of a book need more time to purchase the book, read it, and present their ratings. In general, it is evident from Figure 5.6 that most of the δ values fall in the interval $[0,3]$.

Based on the above observations, we incorporate the average users' ratings for $\delta \in [0,3]$ days as a feature in our affinity rank prediction model. If we denote the upper bound of δ as ℓ, then this feature can be computed as $\overline{\mathbf{R}}_i^{t-\delta-1}$ over all $\delta \in [0,\ell]$.

5.3 Affinity Ranks Prediction

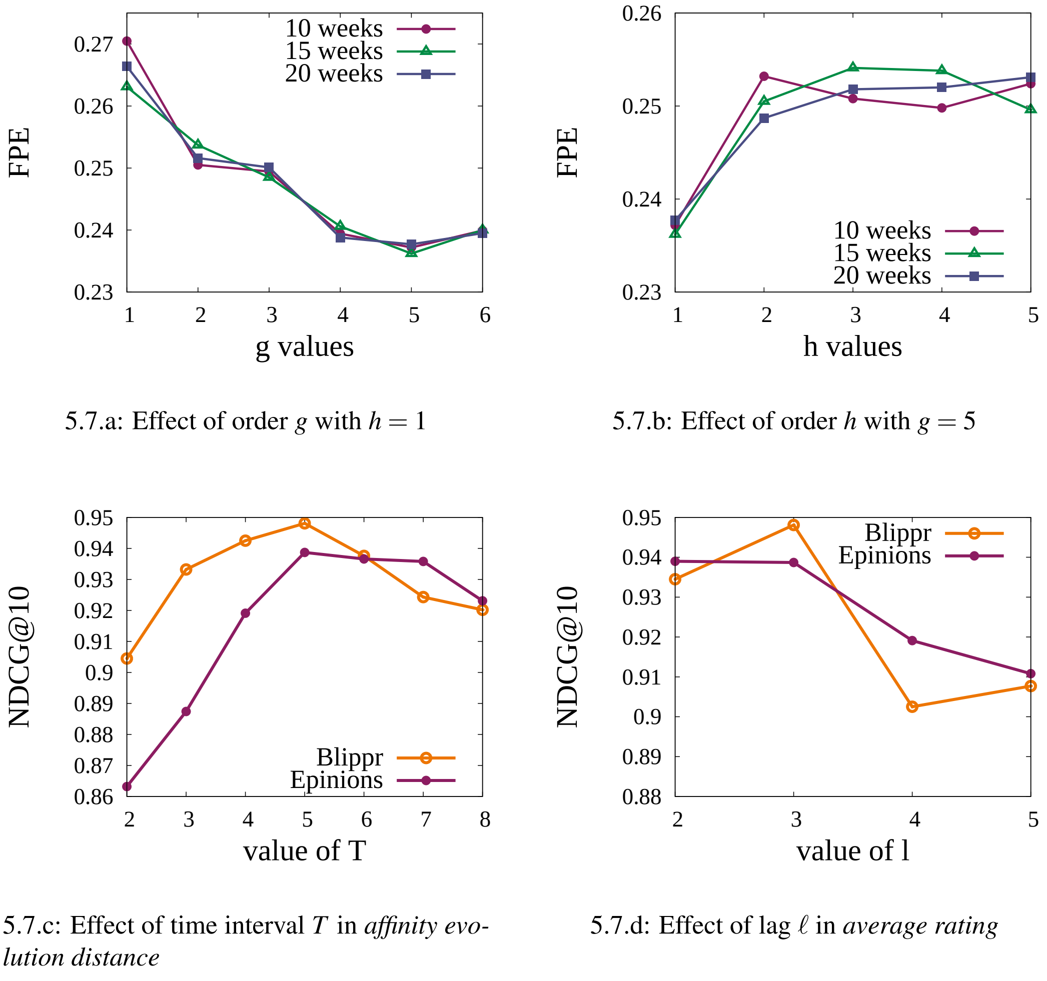
In this section, we propose the *AffRank* model in detail and present an algorithm to predict the affinity ranks.

It is evident from our earlier discussions that the affinity rank of a product is highly related to its ranks in the near past. However, how long of the past history should be taken into account is unknown yet. Thus, SVM-based regression technique (and its variants) cannot be adopted as it is difficult to define data samples with unknown dimensions. Instead, ARX (AutoRegressive model with exogenous inputs) is best suitable for this case as discussed in [121]. ARX model is capable of incorporating external inputs and is widely used in modeling various types of natural and social phenomena [121]. More importantly, it can find the best length of the period that should be taken into account by simply varying the *order* parameter in the model. Additionally, as ARX is a linear model, the weight of each feature clearly indicates how important that feature is. We can simply find from the features which are important and which can be ignored.

The ARX model of orders g and h is given in the following equation.

$$y_t = \varepsilon_t + \sum_{i=1}^{g} \varphi_i y_{t-i} + \sum_{j=1}^{s} \sum_{i=1}^{h} w_{i,j} b_{t-i,j} \qquad \text{(Eq. 5.9)}$$

In this equation, y is the time series data (*i.e.,* product ranks in various different time slots), s is the number of exogenous input features; $\varphi_1,\ldots,\varphi_g$ and $w_{1,1},\ldots,w_{h,s}$ are the parameters to be estimated from the training data. Both g and h are the orders in the model to be manually



5.7.a: Effect of order g with $h = 1$

5.7.b: Effect of order h with $g = 5$

5.7.c: Effect of time interval T in *affinity evolution distance*

5.7.d: Effect of lag ℓ in *average rating*

Figure 5.7: (a)(b)ARX model order optimization; (c)(d)Feature parameter setting.

determined before model estimation. In our context, g is the *order of product rank* which determines the number of previous product ranks to be considered in the modeling; h is the *order of feature* determining the number of past time slots from which the values of the corresponding features to be involved in the model estimation. The variable $b_{t-i,j}$ is the value for feature j at time $t-i$, and ε_t is white noise. The estimation of the ARX model is efficient as it solves linear regression equations in analytic form. Also, the solution is unique and always satisfies the global minimum of the loss function [121].

In the ARX model, the order of product rank g and the order of feature h need to be manually

Algorithm 7: Affinity rank prediction.

```
Input: ARX model parameters $\{\varphi_1, \varphi_2, \ldots, \varphi_p\}$, $\{w_{1,1}, w_{1,2}, \ldots, w_{h,s}\}$, test dataset $\Psi_i^t$ for $p_i \in \mathbb{P}$
Output: the affinity rank $r_t(p_i)$ for all products $p_i \in \mathbb{P}$ at time $t$
1 begin
2     while $j < \mathbb{P}.size$ do
3         compute $y_t^i$ according to Equation Eq. 5.9 using $\Psi_i^t$;
4         $Y[j] = y_t^i$;
5         $j++$;
6     sort($Y$) by ascending order;
7     forall the $Y[j] = y_t^i$ do
8         $r_t(p_i) = j + 1$;
```

determined [6]. A common way of selecting g and h is to fix the value of g (or h) and try a range of values for h (or g); for each pair of g and h values, evaluate the accuracy of the model estimated using a measure called FPE [10] . A smaller FPE value means a more accurate model. Figures 5.7(a) and (b) report the FPE measures of varying values of g while fixing $h = 1$, and varying h while fixing $g = 5$, respectively. Based on the FPE measures, we set $g = 5$ and $h = 1$ for our experimental study (Section 5.4). Observe that in Figures 5.7(a) and (b), the size of the training data (*e.g.,* 10, 15, 20 weeks) has marginal impact on the accuracy of the estimated model for a given pair of g and h values. Hence in our experimental study, we use the latest 10 weeks data to estimate the ARX model.

We now present the algorithm to predict the affinity ranks using the learned ARX model. Detailed in Algorithm 7, for each product, the value of each feature j discussed in Section 5.2 at time $t-i$ (denoted by $b_{t-i,j}$) is derived and stored in feature vector Ψ_i^t. The target value y_t is predicted by the model using Ψ_i^t.

5.4 Experiments

In this section, we report experimental results on *Blippr* and *Epinions* datasets. On *Blippr*, the experiments are conducted on all 75 products with statistics reported in Table 5.2. On

[10]More details of FPE measure can be found in [122].

Epinions, a subset of 1,311 products, each of which received at least 200 ratings, is used in our experiments. There are in total 343,154 ratings in this subset. The goals of the evaluation were to establish whether the proposed *AffRank* model can reliably predict product affinity ranks; compare the performance of the *AffRank* model and three baseline models; and seek to understand the importance of various features in the proposed model.

We begin by reporting the performance metric used to evaluate the accuracy of the predicted product affinity ranking.

Performance Metric. To evaluate the accuracy of the predicted product rank against the ground-truth rank at a given time slot, we adopted Normalized Discounted Cumulative Gain (NDCG) [123] measure which is commonly used in Information Retrieval (IR) to evaluate the effectiveness of ranking algorithms for Web search and other related applications. It is defined in the following equation.

$$NDCG@k = \frac{1}{Z}\sum_{i=1}^{k}\frac{rel_i}{\log_2(i+1)} \qquad \text{(Eq. 5.10)}$$

In the above equation, i is the rank position and k is the number of top-ranked retrieved documents to be considered in NDCG computation. rel_i is the relevance of the ith retrieved document in IR setting. With a perfect ranking, a more relevant document shall be ranked higher than the less relevant document and so on. Lastly, Z is a normalizing constant that ensures the perfect ranking (i.e., the ground-truth rank in our case) achieves $NDCG@k$ of 1.0.

Note that we chose the aforementioned metric for the following reason. In viral marketing and online advertising context, logarithmic decay by positions is a reasonable assumption when products are presented in a list format [124]. For instance, consider the scenario discussed in Section 5.1.1. A company may allocate different investment budget for each movie in the top-5 list. Obviously, more investment should be put into higher ranked movies. Hence, accurate prediction of top ranked movies is much more important compared to lower ranked movies.

Consequently, we use NDCG as it employs a log function to discount the positions and as a result the top positions are considered more valuable than the lower positions.

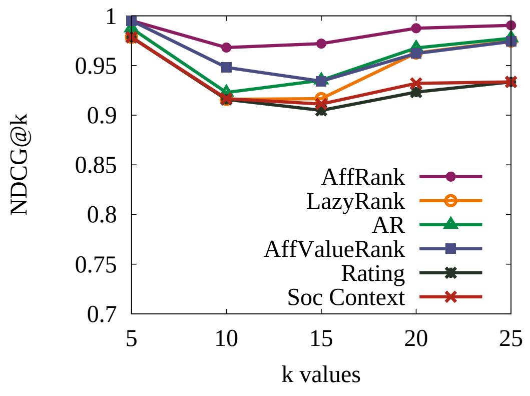
In our experiments, we vary k from 5 to 25 at the step of 5 to evaluate the accuracy of the rank involving top-k ranked products. We define the relevance of a product p_i to be the inverse of its ground-truth rank: $\frac{1}{\rho_t(p_i)}$. Observe that the value of NDCG heavily depends on the definition of relevance (i.e., rel_i). However, for a given relevance definition (e.g., $\frac{1}{\rho_t(p_i)}$), NDCG well reflects the accuracies of different ranking models.

Feature Parameter Setting. Following the approach reported in [6], we now evaluate the impact of the feature parameters. The two feature parameters are the number of time slots T involved in the computation of *affinity evolution distance* (see Section 5.2.3), and the time lag ℓ in computing *average rating* (see Section 5.2.4). With the ARX model fixed with g=5 and h=1, we evaluate the impact of T while fixing ℓ=3. As reported in Figure 5.7(c), the ranking model achieves the best performance when $T = 5$. Similarly, we fix $T = 5$, and report the effect of varying ℓ in Figure 5.7(d). On *Blippr dataset*, the best performance is achieved when $\ell = 3$; On *Epinions dataset*, the best performance is when $\ell = 2$ followed by a marginal drop in performance when $\ell = 3$. On both datasets, the performance drops significantly when ℓ is greater than 3. Hence, in the sequel we set $T = 5$ and $\ell = 3$.

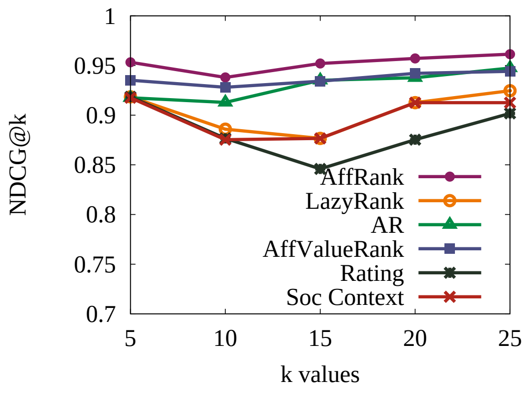
Comparison to Baseline Methods. In this section, we compare the performance of the proposed *AffRank* with three other methods.

LazyRank. This model predicts product rank at time slot t to be the same as the rank obtained in the last time slot $t-1$.

AR. *AR* model refers to the AutoRegressive model without taking exogenous input features (see Equation Eq. 5.9). In another word, the product rank y_t is predicted solely by the ranks in the past g time slots, y_{t-g} to y_{t-1}, for a given product rank order g.



5.8.a: Blippr



5.8.b: Epinions

Figure 5.8: Comparison with other models.

AffValueRank. With *AffValueRank*, instead of predicting the affinity rank, the exact affinity value is predicted using the ARX model. That is, y_{t-i} in Equation Eq. 5.9 is set to be the affinity a_i^{t-i}. Using the learned model, the affinity values a_i^t is predicted; the products are then ranked accordingly.

As reported in Figure 5.8, the proposed *AffRank* model outperforms all three baseline models on both datasets for every k value. Overall, *AffValueRank* is the second best performing model followed by *AR*. The *LazyRank* model performs the worst. Considering the features involved in the four models, the experimental results show that, (i) product affinity rank can be better predicted using the past few product ranks than the single last rank (i.e., *AR* $>$ *LazyRank*), (ii) the extra features besides the product rank lead to better prediction (i.e., *AffValueRank* $>$ *AR*), and (iii), product affinity ranks can be better predicted than the affinity values (i.e., *AffRank* $>$ *AffValueRank*) probably due to the smoother distribution of product affinity ranks than affinity values (see Section 5.2.2).

We also tested two other baseline methods: one is to rank products by actual user rating (*i.e., Rating*) and the other is to rank products by the *social context* feature (*i.e., Soc Context*).

Table 5.8: Top 10 popular products in weeks 12 & 13 (2009) and predicted ranks. (Blippr)

Rank	*Week 12 (LazyRank)*	*Week 13*	*Predicted rank in week 13*		
			AffRank	*AR*	*AffValueRank*
1	Twitter	Twitter	Twitter	Twitter	Twitter
2	Gmail	Gmail	Gmail	Gmail	Gmail
3	Mashable	Mashable	Mashable	Mashable	Mashable
4	Google	Tweetie for iPhone	Google Earth	Google Earth	Google
5	Google Earth	Google Earth	Google	Google	Google Earth
6	OK Computer	Google	Tweetie for iPhone	Google Reader	The Dark Knight
7	Google Reader	The Dark Knight	The Dark Knight	The Dark Knight	Tweetie for iPhone
8	Dropbox	The Shawshank Redemption	The Shawshank Redemption	Tweetie for iPhone	The Shawshank Redemption
9	WordPress	Google Reader	Google Reader	WordPress	Google Reader
10	In Rainbows	Watchmen	Watchmen	OK Computer	OK Computer

However, none of these approaches can outperform the simplest *LazyRank* model as shown in Figure 5.8.

Case Study. Table 5.8 shows an example of predicted affinity rank in *Blippr*. The second and third columns show the top-10 ranked products in weeks 12 and 13 of year 2009, respectively. The remaining 3 columns list the predicted ranks using different models. Obviously, our model *AffRank* accurately predicts the top 10 products in week 13 although the exact ranks for products *Google Earth*, *Google* and *Tweetie for iPhone* are not accurate. Besides, our model also detects that *Google Earth* will acquire more affinity than *Google* in week 13. Compared to week 12, there are four products newly listed in top-10 (i.e., *Tweetie for iPhone*, *The Dark Knight*, *The Shawshank Redemption*, and *Watchmen*). Our *AffRank* predicted all four accurately, *AffValueRank* missed one of them, *AR* missed two of them, and the *LazyRank* missed all four.

Affinity Feature Comparison. In the learning of the ARX model, all feature values were normalized into the same range of $[0,1]$. Hence, the larger the learned weights (e.g., φ_i and $w_{i,j}$ in Equation Eq. 5.9) in the ARX model the more strongly it influences product affinity ranking. Table 5.9 reports the average learned weights over both datasets. Observe that besides *affinity rank history*, *affinity evolution distance* has the biggest weight value which indicates that it plays an important role in the model. As *average rating* feature has a negative weight value, it

Table 5.9: Learnt weights of features with $g = 5, h = 1$.

Feature	*Weight* φ, w in Equation Eq. 5.9
Aff. Rank History (φ_1 - φ_5)	$[0.720, 0.295, 0.267, 0.131, 0.113]$
Aff. Evolution Distance	0.210
Average Rating	-0.124
Community Size	0.021
Social Context	0.007
Member Connectivity	-0.013

suggests that a larger rating leads to a higher rank. Also observe that the *affinity rank history* (φ_1 to φ_5) follows a decreasing tend, suggesting that the more recent product ranks have more impact on its future rank. However, the three traditional features, listed in the bottom part of the table, all have very small weights indicating their negligible impact on product affinity ranking.

In summary, in social rating networks, community size, member connectivity, and social context do not have significant impact on the affinity of a product. Instead, growth of a product community is mainly because of its high affinity ranks in the past few weeks (which may make the product reach larger audience by appearing in the top ranked list on the website's homepage) and the received good ratings from users. Besides, a product showing similar affinity intensity evolution pattern with the most popular product is also likely to be popular in near future.

5.5 Summary

In this chapter, we analyzed two publicly-available social rating networks and proposed a predictive model called *AffRank*, that utilizes an array of features to predict the future rank of products according to their affinities. Informally, product affinity refers to a product community's ability to "attract" new members and is measured by the number of new ratings during a specific time slot. Such information plays an important role in several real-world applications such as online advertisement and marketing research.

We formulate the product affinity prediction problem as an autoregressive model with exogenous inputs (features). We have identified in total six features, namely community size, member connectivity, social context, affinity rank history, affinity evolution distance, and average rating, for predicting product affinity. Our investigation revealed that the community size, member connectivity, and social context do have negligible influence on product affinities. Instead, the remaining features are the most important factors affecting future rank of products. Specifically, we discovered several interesting findings related to these features which we exploit in our model. Firstly, affinity of a product for most products tends to increase spikily and decrease smoothly. Secondly, the average DTW distances between products in the same category are always smaller than those between products from different categories. Thirdly, we studied the lag between the peak in the average rating curve and that in the affinity intensity curve. The affinity intensity is highly correlated with the users' average ratings. Particularly, as the average rating increases the affinity intensity increases accordingly within 3 days. Our exhaustive experimental study demonstrates the effectiveness and superior prediction quality of *AffRank* compared to three baseline methods.

Chapter 6

PATINA: A Partitioning-based Efficient Strategy for Influence Maximization

In the previous chapters, we discussed our work with respect to individual-level and community-level social influence study. In this chapter, we shall present our work called PATINA (a Partitioning-based Efficient Strategy for Influence Maximization in Large Online Social Networks) with respect to network-level social influence study. The rest of the chapter is organized as follows. In Section 6.1 we briefly introduce the influence maximization problem and the limitations of the state-of-the-art technique. We formally define the partitioning-based influence maximization problem in Section 6.2. In Section 6.3 we introduce the structure of MAG-list. The PATINA algorithm is presented in Section 6.4. We discuss how PATINA can be adopted on a distributed framework in Section 6.5. Experimental results are presented in Section 6.6. Finally, the last section concludes the chapter. The notations used in this chapter are summarized in Table 6.1.

6.1 Introduction

Consider the following scenario as a motivating example. Suppose a company wants to market a new product through the social network of potential customers as depicted in Figure 6.1(a). This network consists of eight individuals represented by nodes and the edges between them

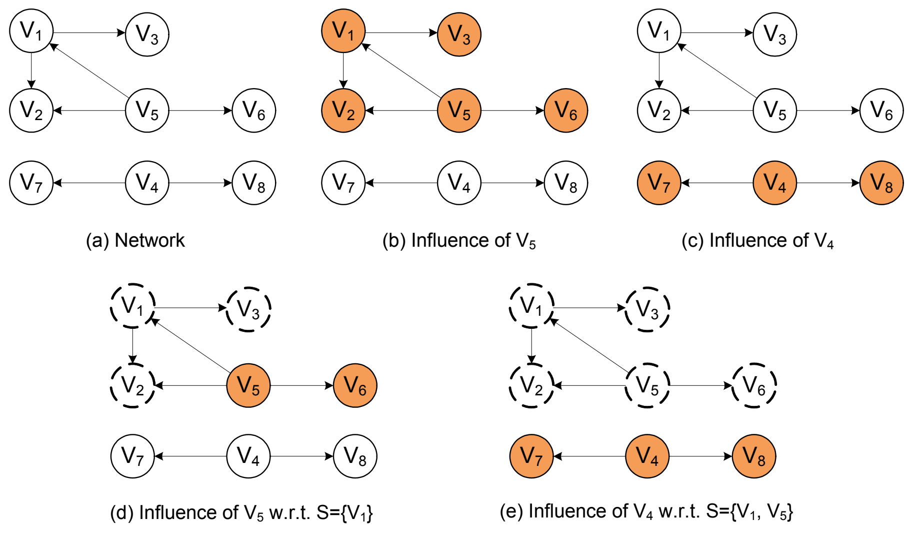

V1
V2
V3
V4
V5
V6
V7
V8
(a) Network
(b) Influence of V5
(c) Influence of V4
(d) Influence of V5 w.r.t. S={V1}
(e) Influence of V4 w.r.t. S={V1, V5}

Figure 6.1: Influence maximization problem.

represent connections or relationship between individuals. The company has a limited budget such that it can only select a small number of initial users (significantly less than 8 in Figure 6.1(a)) in the network to use it for free. It hopes that these users will like the product and trigger a cascade of "word-to-mouth" influence by recommending the product to other friends, and many individuals will ultimately adopt it. Obviously, the company expects the size of the influenced population should be as large as possible. Given a social network as well as an influence propagation model, the problem of *influence maximization* is to find the set of initial users of size k (referred to as *seeds*) so that they eventually influence the largest number of individuals (referred to as *influence spread*) in the network [4]. For instance, Figures 6.1(b)-(e) depict the influence spreads for different seed sets. The colored circles denote the nodes that can be influenced by the specified node while dashed circles denote the nodes that are already influenced by current seeds S.

Due to its importance, there is a long history of study related to the influence of new products and innovations in the social sciences. The work in [125, 126] carried out systematic investigations related to the adoption of medical and agricultural innovations in developing and developed countries. Researchers have also studied the diffusion processes for "word-of-mouth" and "viral marketing" effects in the success of new products [127, 3, 128, 64].

6.1.1 Motivation

Domingos and Richardson [64, 66] are the first to study influence maximization as an algorithmic problem. They proposed a probabilistic model of interaction and heuristics were given for choosing customers with a large overall effect on the network. Kempe et al. [4] are the first to consider the problem of choosing the seeds as a discrete optimization problem. Influences are propagated in the network according to a stochastic cascade model. They proved that the optimization problem is NP-hard, and presented a greedy approximate algorithm that is applicable to three cascade models, namely the *independent cascade* model, the *weighted cascade* model, and the *linear threshold* model. The proposed algorithm has two key strengths. First, it guarantees that the influence spread is within $(1-1/e)$ of the optimal influence spread where e is the base of the natural logarithm. Second, it outperforms the classical degree and centrality-based heuristics in the influence spread.

Despite the benefits of the aforementioned approaches, a key limitation is their efficiency and scalability. It could take days on a modern server machine to find a small seed set in a medium-sized network (e.g., 15,000 nodes) [42]. Consequently, deployment of these techniques on large-scale social networks is not feasible. Recently, several approaches [42, 43] are proposed to address this issue. While they have been able to make significant progress in reducing the computation cost of the influence maximization problem, the most recent greedy approximation technique [42] still requires more than 14 hours to find seed set of size 100 in a network with only 37,154 nodes (see Section 6.6 for details). Given the existence of real-world networks containing millions of nodes (e.g., *Wiki-talk*), it will take days by these greedy approaches to find seeds. In summary, the state-of-the-art greedy algorithms for influence maximization still suffer from the limitation of high computation cost and scalability.

To alleviate the aforementioned performance bottleneck, Chen et al. [42] proposed *degree discount* heuristics that exploits degree information to bring in tremendous improvement to the computation time. Specifically, this heuristic-based technique is orders of magnitude faster than all greedy algorithms. However, the quality of seed set with respect to influence spread can be inferior compared to greedy approaches for many networks. Particularly, as we shall see in Section 6.6, the degree discount-based technique finds seed set that only influences 60% of the size of influence spread computed by greedy approaches. It is worth mentioning that although the importance of reduction in computation time is undeniable, the seed set quality is more significant to companies as ultimately they would like to maximize the influence spreads of their new products in order to reach out to largest possible customers base. *Is it possible to design a greedy approach that can significantly reduce the computation time for seed set without compromising on the quality of influence spread?* In this chapter, we provide an affirmative answer to this question.

6.1.2 Overview and Contributions

We propose a novel greedy algorithm called PATINA (**PA**rtitioning-based **T**echnique for **IN**fluence m**A**ximization) that somewhat depart from existing influential maximization techniques in the following way: where existing strategies essentially consider the network as a whole, PATINA focuses on partitioning the network into a set of non-overlapping *components* (subnetworks) and *distribute* the influence maximization computation to these components. A node's influence in one component is not significantly affected by nodes in other components. Consequently, each node's influence computation and updates can be limited to the component it resides. If m and m' are the number of edges in the entire network and that in the largest component, respectively, and R represents the number of iterations then the worst-case time complexity of PATINA is $O(km'R)$ for solving the influence maximization problem. Note that the most recent greedy technique for influence maximization technique [42] has a complexity of $O(kmR)$. Thus, PATINA is m/m' times faster compared to [42]. Observe that as $m' \ll m$ in many real-world networks [129, 130, 131], the performance gain of PATINA is substantial.

Specifically, as we shall see in Section 6.6, PATINA can find seeds in networks containing millions of nodes in several hours whereas the greedy technique in [42] takes days. More importantly, unlike heuristic-based techniques, performance gain of PATINA is achieved without compromising on the superior quality of seed set.

The aforementioned strategy is motivated by several studies that have shown that many large social networks are comprised of series of communities and clusters where a piece of information can easily spread within the community but hard to propagate from one to another [5, 132, 133, 129, 130, 134, 135]. For instance, the most recent research [5] has shown that in blogosphere, the connections between posts written in different languages is so limited that we can almost ignore them. Thus, the network can be separated into several subnetworks based on language barriers.

A key challenge in our proposed approach is to determine the subnetworks from which the seeds need to be selected. To address this issue, we present an efficient data structure called MAG-*list* (**MA**rginal **G**ain List), which stores the *candidate node* having maximum *marginal gain* from each component in the network and guides us to determine the members of seed set. MAG-list is space-efficient as it only requires $O(\ell)$ space complexity, where ℓ is the number of partitioned subnetworks. Thus, in contrast to existing greedy approaches, we do not need to keep the entire collection of nodes of the network in the memory. Additionally, it provides an efficient framework to update the influence of nodes. Note that whenever a node is selected into the seed set, some other nodes' influence may change as well. Thus, it is important to dynamically update the influence of each node. Particularly, PATINA applies an *on-demand update* strategy in each round to update the MAG-list. Only when a node in the MAG-list is selected as a potential candidate for the seed set, PATINA updates all the nodes in the *component gain sublist* (COG-sublist) of this node. It is not necessary to update all nodes in the MAG-list. Consequently, we avoid many unnecessary updates.

PATINA differs from the aforementioned approaches in the following key ways. Firstly, we partition the network into a set of non-overlapping subnetworks and distribute the influence maximization problem to relevant subnetworks to compute the seed set. Note that the time and space complexities of PATINA reduce significantly as it runs on subnetworks which are often significantly smaller in sizes compared to the entire network. In contrast, as existing techniques are designed to take the entire network as input for influence maximization, all the greedy approaches result in high computation cost due to the gigantic size of many online social networks. Secondly, due to the partitioning of the network into subnetworks, PATINA needs to determine efficiently the subnetworks that will contribute to the seed set. Such *seed selection* problem did not exists in existing approaches. Thirdly, in comparison to the degree discount approach, PATINA can find significantly better quality of seeds. As we shall see in Section 6.6, on networks containing millions of nodes, for a given budget, the degree discount approach can find seeds that can potentially influence around 33,000 individuals. In contrast, PATINA can find seeds that can influence more than 55,000 individuals. Given the fact that companies may invest months or years in designing new products, it is paramount to find seeds that give them opportunity to influence a larger population. We believe that companies are willing to wait few hours to find higher quality seed set as it may have significant impact on the marketing of products and its profits.

Lastly, a key benefit of our proposed approach is that it can not only be realized on a single machine but also it can easily be adopted on a distributed platform. Specifically, the MAG-list can be maintained in a central machine and the maximization of influence for the subnetworks can be distributed into several machines and computed in parallel. This further reduces the computation time of seed set. In summary, the key contributions of this chapter are the followings.

- We present a novel data structure called MAG-*list* which facilitates efficient computation of influence maximization problem. It also provides an efficient framework to support updates of nodes' influences.

- Departing from existing centralized, "non-partitioning-based" solutions to the influence maximization problem, *we take the first step to propose a novel approach that addresses this problem by partitioning the underlying network into a set of non-overlapping subnetworks using an existing network partitioning technique and distributing influence spreads computation to relevant subnetworks.* Specifically, we present a novel partitioning-based greedy algorithm called PATINA that efficiently exploits the MAG-list build on top of the partitioned subnetworks to compute the seed set for influence maximization while guaranteeing superior quality of the influence spread. Importantly, PATINA is at least m/m' times faster than the state-of-the-art greedy techniques. Further, it produces superior quality seed set compared to heuristic-based techniques. Additionally, due to its inherent characteristics the PATINA framework can be gracefully adopted on a distributed platform, which can improve its running time substantially.
- By applying PATINA to real-world social networks of various sizes, we show its effectiveness, significant improvement of performance over existing methods, and ability to work on a distributed platform.

6.2 Problem Statement

In this section, we formally define the *partitioning-based influence maximization* problem. We begin by briefly describing the classical influence maximization problem and related concepts.

6.2.1 Classical Influence Maximization Problem

A social network is modeled as graph $G = (V, E)$, where nodes in V modeling the individuals in the network and edges in E modeling the relationship between the individuals. We use n to denote the number of nodes and m to denote the number of edges throughout the paper. The influence maximization problem is defined as follows [4].

Symbol	Definition
$G(V,E)$	A social network graph
n	number of vertices in G
m	number of edges in G
$G_i(V_i,E_i)$	i^{th} component (subnetwork) in $G(V,E)$
Φ	A set of subnetworks (components)
m'	$\max\limits_{E_i \in E} \lvert E_i \rvert$
k	number of seeds to be selected
ℓ	number of connected components (subnetworks)
R	number of rounds of simulation
β^i	A COG-sublist
Ω	A set of COG-sublists
$\mathcal{M}$	MAG-list
S	seed set
S_i	seed nodes selected from $G_i(V_i,E_i)$
p	propagation probability in independent cascade model
$\sigma_i(\cdot)$	influence function under cascade model C_i
T	number of iterations in gain computation under weighted cascade model [42]

Table 6.1: Notations.

Definition 6.15 [Influence Maximization] *Given a social network $G(V,E)$, a specific cascade model C and a budget number k, the* ***influence maximization problem*** *is to find a set of nodes S in G, which we call as seed set, where $|S| = k$ such that according to C, the expected number of nodes that are influenced by S (denoted by $\sigma(S)$) is the largest. It can be expressed using the following formula.*

$$S = \operatorname*{arg\,max}_{S' \subseteq V, |S'|=k} \sigma(S')$$

In the above definition, cascade model refers to the model that defines how a piece of information propagates from an individual to another in the network. Existing research have focused on the following cascade models as defined in [4].

- *Independent cascade (IC) model.* Let A_i be the set of nodes that are influenced in the i-th round and $A_o = |S|$. For any $(u,v) \in E$ such that u is already in A_i and v is not yet influenced, v is influenced by u in the next $(i+1)$-th round with an independent probability p, which is referred to as the *propagation probability*. Thus, if there are t neighbors of v that are in A_i, then $v \in A_{i+1}$ with probability $1-(1-p)^t$. This process is repeated until A_{i+1} is empty.

- *Weighted cascade (WC) model.* Let $(u,v) \in E$. In this model, if u is influenced in round i, then v is influenced by u in round $(i+1)$ with probability $1/v.degree$. Thus, if v has t neighbors influenced at the i-th round then the probability for a node v to be influenced in the next round is $1-(1-1/v.degree)^t$. It is sometimes classified as an instance of IC model [4].

- *Linear threshold (LT) model.* In this model, each node v has a threshold θ_v uniformly and randomly chosen from 0 to 1; this represents the weighted fraction of v's neighbors that must become influenced (active) in order for v to be influenced. All nodes that were influenced in step $(i-1)$ remains so in step i, and any node v is influenced when the total weight of its influenced neighbors is at least θ_v.

The optimum solution to the influence maximization problem is NP-hard for the aforementioned cascade models [4]. However, as discussed in the preceding section, greedy approximation algorithms exist for the optimal solution to be approximated to within a factor of $(1-1/e)$ as long as the influence function $\sigma(\cdot)$ is *submodular*. Let S be a finite set. Then a function $f: 2^S \to R$ is *submodular* if $f(A \cup \{v\}) - f(A) \geq f(B \cup \{v\}) - f(B)$ $\forall A \subseteq B \subseteq S$ and $v \in S$. In another word, the *marginal gain* from adding an element to a set A is at least as much as the *marginal gain* from adding the same element to a superset of A. In the case of influence maximization problem, $\sigma(\cdot)$ is submodular, takes only nonnegative values, and is monotone in the sense that adding an element to a set cannot cause f to decrease. The *marginal gain* of a node v given the seed set S is defined as following [4].

Definition 6.16 [Marginal Gain] *Given a cascade model C, a node v, and the current seed set S, the* ***marginal gain*** *of v with respect to S, denoted by $\sigma(v|S)$, is defined as $\sigma(v|S) = \sigma(S \cup \{v\}) - \sigma(S)$. That is, $\sigma(v|S)$ denotes the increase in the expected number of nodes that is influenced due to the addition of v in S.*

Greedy solution towards influence maximization problem works by iteratively selecting the node which shows the most marginal gain for current S. Thus, each time after adding a node into S, the greedy algorithm has to update each node's marginal gain for current S and select the one with the maximum marginal gain. For example, consider Figure 6.1. The greed solution first selects v_5 as the first seed as it has the maximum marginal gain when $S = \varnothing$. Then, it selects v_4 as the second seed as the marginal gain of other nodes are all zero given that $S = \{v_5\}$. Obviously, each time after selecting a seed, the greedy algorithm needs to recompute the marginal gain for every node for the new seeds S, which is computationally expensive. Observe that the marginal gain of v_4 is not affected (Figure 6.1(e)) for $S = \{v_1\}$ or $S = \{v_5\}$. Hence it is not necessary to update the marginal gains of v_4, v_7, or v_8 when v_1 or v_5 is selected as seed. Similarly, if v_4 is selected as seed, the marginal gains of v_1 and v_5 do not change either.

6.2.2 Partitioning-Based Influence Maximization Problem

Existing greedy approximation algorithms consume significant time on updating the marginal gains of the top nodes in the list and their rearrangements [4, 42, 43]. Hence, avoidance of unnecessary updates of marginal gains along with reduction of the size of the node list can reduce the computation cost significantly. In this work, we achieve this by taking a partitioning-based approach where the whole social network is partitioned into a set of non-overlapping subnetworks. By doing so, we ensure that changes to the marginal gain of a node in a subnetwork G_i do not affect nodes in another subnetwork G_j. Hence, the update of the marginal gains of nodes in G_i is restricted within it instead of the entire network. In fact, as we shall see later, the

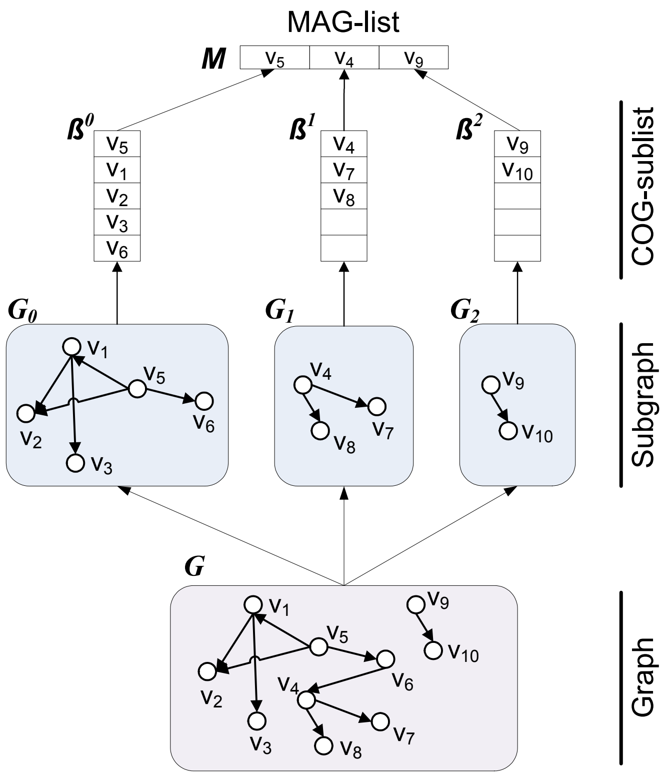


Figure 6.2: The structures of MAG-list and COG-sublists.

computation time of the update operation is reduced by a factor of m/m_i where m_i represent the number of edges in G_i.

Definition 6.17 [Partitioning-based Influence Maximization Problem] *Given a budget number k and a social network graph $G(V,E)$, let $\Phi = \textbf{Partition}(G)$ be the partitions of G containing a set of subnetworks where $V = V_1 \cup V_2 \cup \ldots \cup V_{|\Phi|}$, $V_i \cap V_j = \varnothing$ for $\forall\, i \neq j$, $0 \leq (i,j) < |\Phi|$, and $(u,v) \notin E$ for $\forall\, u \in V_i, v \in V_j$. Let each G_i exhibits a specific cascade model C_i. Then the **partitioning-based influence maximization problem** finds a set of seeds S in Φ where $|S| = \sum_{i=1}^{|\Phi|} |S_i| = k$ such that the expected number of nodes that are influenced by S is the largest in G. That is,*

$$S = \underset{\sum |S_i| = k}{\arg\max} \sum_{S_i \subseteq V_i} \sigma(S_i)$$

Note that it is important for the partitioning-based influence maximization problem to guarantee that the influence spread is within $(1-1/e)$ of the optimal influence spread.

Theorem 6.2 *Given the social network graph $G(V,E)$ and $\Phi = \textbf{Partition}(G)$, if the influence function $\sigma_i(\cdot)$ for each of the cascade model C_i of $G_i \in \Phi$ is submodular, then $\sigma(S)$ in Definition 6.17 is also submodular.*

Proof: *(Sketch)* According to Definition 6.17, $\sigma(S)$ can be represented as the following.

$$\sigma(S) = \max_{\sum|S_i|=k} \sum \sigma_i(S_i)$$

Assume $S' \subset S, v \in V_t \setminus S_t$ where $t \in \{1 \ldots \ell\}$ and $S = S_1 \cup S_2 \cup \ldots \cup S_\ell, S' = S'_1 \cup S'_2 \cup \ldots \cup S'_\ell$, then $S'_i \subseteq S_i$. Besides, the following expression holds as $S_i \cap S_j = \varnothing \; \forall \; 0 < (i,j) \leq \ell$.

$$\begin{aligned} \sigma(S \cup \{v\}) - \sigma(S) &= \sigma_p(S_t \cup \{v\}) - \sigma_t(S_t) \\ \sigma(S' \cup \{v\}) - \sigma(S') &= \sigma_p(S'_t \cup \{v\}) - \sigma_t(S'_t) \end{aligned}$$

As $S'_t \subseteq S'$ and the influence function $\sigma_t(\cdot)$ is submodular, then $\sigma_t(S_t \cup \{v\}) - \sigma_t(S_t) \leq \sigma_t(S'_t \cup \{v\}) - \sigma_t(S'_t)$ holds according to the definition of submodularity. Thus, the following expression holds too, which means that the influence function $\sigma(S)$ is submodular.

$$\sigma(S \cup \{v\}) - \sigma(S) \leq \sigma(S' \cup \{v\}) - \sigma(S')$$

∎

Observe that the above theorem states that if the influence functions within each partition are submodular, then we have the usual $(1-1/e)$ guarantee for the solution quality for the partitioned network. In particular, in the networks which contains disconnected components we can easily apply BFS (Breadth First Search) method to obtain the partitions. The partitions retrieved using this methods do not have any connection with each other, thus will be in accord with the condition in Def 6.17 (*i.e.,* $(u,v) \notin E$ for $\forall \; u \in V_i, v \in V_j$). In the networks where BFS cannot identify a series of partitions, we apply min cut-based methods and thus may not be strictly in accord with this condition. Hence, it does not indicate that the partitioning-based solution will have the $(1-1/e)$ guarantee for the original optimization problem defined on the whole network. In spite of this, as we shall see later, our empirical results on variety of real social networks demonstrate that PATINA, even using min cut-based partitioning method, consistently produces superior quality spreads compared to the conventional greedy approaches having $(1-1/e)$ guarantee [4].

6.2.3 Overview of PATINA

In this section, we provide an overview of the PATINA algorithm designed to address the partitioning-based influence maximization problem. It is outlined in Algorithm 8 and consists of three phases, namely the *network partitioning* phase (Line 2), the MAG-*list construction* phase (Line 3), and the *seeds selection* phase (Line 4).

The network partitioning phase. For any cascade model, influence always flows along edges in the social network graph. Hence, if there is no path between two nodes then it is not possible for influence to flow between these nodes. In this phase, we first partition the social network graph to a set of non-overlapping connected components (also referred to as subnetworks). As each component is unconnected to another component, the influence computation in a subnetwork is not affected by other subnetworks or components.

Note there are several existing techniques to generate disjoint dense connected components from a graph efficiently [136, 137]. We take the BFS (Breadth First Search)-based strategy to traverse the graph and extract the connected components. The running time of this process is $O(m+n)$. Note that if the network can be split into disjoint components using BFS technique where $m' \ll m$ (e.g., "High Energy Physics - Theory" collaboration network in *arXiv* [1]) then the subsequent computation of seed set generates seeds that are of similar quality of

[1]http://www.arXiv.org

Algorithm 8: The PATINA algorithm.

```
  Input: Graph $G(V,E)$, budget $k$ and the cascade influence function $\sigma(\cdot)$
  Output: Seed set $S$ of nodes, $|S| = k$
1 begin
2     $\Phi \leftarrow$ NetworkPartition($G$);
3     $(\mathcal{M},\Omega) \leftarrow$ MAGConstruction($\Phi$, $\sigma(\cdot)$) /* Algorithm 9 */;
4     $S \leftarrow$ SeedsSelection($G$, $k$, $\sigma(\cdot)$, $\mathcal{M}$, $\Omega$) /* Algorithm 10 */;
```

those produced by "partition-free" greedy strategies (e.g., MixGreedy [42]). However, some real-world networks (e.g., *Wiki-talk*) are highly clustered and cannot be easily separated into a set of non-overlapping subnetworks using the BFS technique. Additionally, the BFS-based method may generate components having $m' \approx m$ for these networks. In this case, we partition the network into non-overlapping components using a ℓ-way partitioning algorithm provided by CLUTO [2] [136]. Given the number of partitions ℓ as input, the algorithm can provide good quality partitions in $O(m)$ time. Note that such partitioning process may inevitably remove some edges in the network. However, as graph partitioning algorithms often minimize the size of edge cuts, the removal of edges does not have significant adverse effect on the estimation of influences of nodes in comparison to existing greedy approaches. Consequently, our experimental results in Section 6.6 demonstrate that for these networks PATINA can still preserve high quality seed set, which can influence more than 90% of the size of influence spread computed by "partition-free" greedy approaches.

In summary, we undertake the following strategy for partitioning the social network graph. If the network can be easily clustered into non-overlapping components by BFS-based method such that $m' \ll m$, then we create the final subnetworks based on this strategy. However, if the BFS-based method fails to generate disjoint components or there exists components after partitioning such that $m' \approx m$, then we adopt the ℓ-way partitioning technique to generate the set of non-overlapping subnetworks.

[2] http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview

The MAG-list construction phase. Recall that existing greedy approaches iteratively add to S the node having maximum marginal gain with respect to current S followed by update of each node's marginal gain for S. In contrast to the strategy of lazily updating the marginal gains of nodes existing in a single set, in partitioning-based approach, a set of node sets representing the set of subnetworks exist. Consequently, the update of marginal gains needs to be carried out within each node set independently. Given that there may be a large number of subnetworks, how can we efficiently perform the update operations? In this phase, we construct two data structures, namely MAG-*list* and a set of COG-*sublist*s over the subnetworks, that enable us to efficiently determine which subnetwork the next seed should be selected from and how to effectively perform updates of marginal gains across subnetworks. Informally, a MAG-*list* contains nodes with maximum marginal gain in the subnetworks. Each COG-*sublist i*s associated with a subnetwork or component and stores the marginal gains of all nodes in the subnetwork. We shall elaborate on this phase in Section 6.3.

The seeds selection phase. Lastly, this phase exploits the MAG-list to compute the seed set S from the set of subnetworks. It iteratively selects the node having maximum marginal gain from the MAG-list and, if necessary, efficiently updates and reorders nodes in relevant COG-sublists dynamically. We shall elaborate on it in Section 6.4.

6.3 MAG-List Construction

In this section, we present the MAG-list (**MA**rginal **G**ain List) data structure, which we shall be exploiting for the influence maximization problem. We first describe the structure of the list and then present the algorithm for constructing it.

We begin by introducing the notion of ***component gain sublist*** (COG-sublist) which we shall be using to define MAG-list. Given a subnetwork $G_i(V_i, E_i)$ where $G_i \in \Phi$, the *component gain sublist* of G_i, denoted by β^i, contains the list of nodes V_i. Each node $v \in \beta^i$ and $v \in V_i$ is a 3-tuple $(ID, gain, valid)$ where ID is the unique node identifier of v in G, $gain$ is the marginal

Algorithm 9: The *MAGConstruction* Algorithm.

```
Input: Non-overlapping subnetworks $\Phi = \{G_0(V_0,E_0), G_1(V_1,E_1), \ldots, G_{\ell-1}(V_{\ell-1},E_{\ell-1})\}$ of the
       social network graph $G(V,E)$, the cascade influence function $\sigma(\cdot)$
Output: MAG-list $\mathcal{M}$ and a set of COG-sublists of $\Phi$ denoted by $\Omega$.
1  begin
2      initialize the MAG-list $\mathcal{M}$ of size $\ell$;
3      foreach $G_i(V_i,E_i) \in \Phi$ do
4          initialize COG-sublist $\beta^i$;
5          foreach $v \in V_i$ do
6              $v.valid = 0$;
7              $\beta^i.append(v)$;
8          $\Omega.add(\beta^i)$;
9      for $iter = 1$ to $R$ do
10         for $i = 0$ to $\ell - 1$ do
11             compute $G'_i(V_i,E'_i)$ by removing each edge from $G_i(V_i,E_i)$ with probability $1-p$ (IC
               model) or $1 - 1/v.degree$ (WC model);
12             foreach $v \in V_i$ do
13                 $v.gain += \sigma_i(v)$;
14     for $i = 0$ to $\ell - 1$ do
15         sort($\beta^i$) by $\beta^i.gain$ in descending order;
16         $top(\beta^i).valid = 1$;
17         $\mathcal{M}[i] = top(\beta^i)$;
18     return $(\mathcal{M}, \Omega)$
```

gain with respect to S_i, and *valid* is a boolean variable indicating whether the marginal gain of v is up-to-date. The list is sorted in descending order based on the marginal gains of the nodes. Hence, the node with maximum marginal gain is the top element in the sublist, denoted by $top(\beta^i)$. The *size* of COG-sublist is denoted by $|\beta^i| = |V_i|$. Note that since a social network graph is partitioned into a set of non-overlapping subnetworks, each subnetwork is associated with a COG-sublist.

Informally, a MAG-*list*, denoted by $\mathcal{M}$, contains a list of nodes where each node represents the node with maximum marginal gain in a COG-list. That is, $\mathcal{M}[i] = top(\beta^i)\ \forall\ 0 \leq i < |\Phi|$. Note that the size of $\mathcal{M}$ is the number of non-overlapping subnetworks or components generated from the social network graph G. Figure 6.2 depicts an example of the structures of

COG-sublists and MAG-list.

Definition 6.18 **[MAG-list]** *Given the social network graph $G(V,E)$, let $\Phi = \boldsymbol{Partition}(G)$ where $|\Phi| = \ell$. Then, the* **MAG-*list***, *denoted by $\mathcal{M}$, is a list of nodes of size ℓ where $M[i] = top(\beta^i)\ \forall\ 0 \leq i < \ell$.*

To facilitate the discussions on algorithms, we assume some auxiliary functions of nodes. Given a node v, $append(v)$ and $remove(v)$ append and remove v from a node set or COG-sublist, respectively. Algorithm 9 outlines the MAG-list construction algorithm. For each subnetwork $G_i(V_i,E_i)$ it first initializes a COG-sublist β_i and populates it by setting the $valid$ attributes of the nodes to 0 (Lines 3-8). Next, for nodes in each subnetwork G_i it computes the marginal gains based on a specific cascade model and assign them to the list of nodes in β_i (Lines 9-13). The nodes in β_i are sorted in descending order of their marginal gains (Line 15). We set the valid attributes of all $top(\beta^i)$ to 1 as in the first iteration their marginal gains equal to their influences (Line 16). Lastly, the algorithm constructs the MAG-list $\mathcal{M}$ by inserting the top element $top(\beta^i)$ of each β^i (Line 17). Note that the MAG-list construction requires only a linear traversal over the COG-sublists.

6.4 Seeds Selection

In this section, we discuss the seeds selection phase in detail. Let us illustrate the idea intuitively with the example in Figure 6.2. The MAG-list $\mathcal{M}$ contains the nodes v_5, v_4, and v_9. In the first round of iteration, we select the node having maximum marginal gain from the MAG-list (i.e., v_5) as a candidate. We check if its gain is up-to-date ($valid$ field is 1). Recall from Algorithm 9, the top node in each COG-sublist is marked as valid. That is, the marginal gains of all nodes except the top node in a COG-sublist is set to 0 (not up-to-date). Thus, v_5 is valid in this round. Consequently, we insert it into S and remove it from G and the COG-sublist β^0. Now v_1 moves to the top of β^0 and hence it is copied to $\mathcal{M}[0]$. In the next round, assume that

v_1 is the node with the maximum marginal gain in $\mathcal{M}$ and hence is selected as a candidate. However, v_1's gain is not up-to-date. Consequently, we need to update v_1's gain as it may change due to addition of v_5 in S. The update process works as follows. We recompute the marginal gain of v_1 in β_0 and check whether v_1's gain is still the highest. If it is, then we mark v_1 as valid. Otherwise, we move v_1 to the correct position in the COG-sublist β_0 to ensure that the list remains sorted in descending order. After the update is completed, we select the next candidate for the next round. The seed selection process terminates when there are k nodes in S.

Algorithm 10 outlines the aforementioned intuition for finding the seed set using the MAG-list. It iteratively selects from the MAG-list the node v' having the maximum gain as a candidate (Line 3). Then the algorithm checks whether the v''s gain is updated by evaluating its *valid* field (Line 4). If it is already updated, then it inserts v' into S_r. Next, it removes v' from the COG-sublist β_r as well as G and continue to the next round (Lines 5-8). Otherwise, the candidate node's gain is not up-to-date. Consequently, the algorithm updates v''s marginal gain and reorders β_r by invoking the *update* procedure (Lines 10), which we shall elaborate later. Then it updates $\mathcal{M}[r]$ using the top element $top(\beta_r)$ (Line 11). The algorithm terminates when there are k nodes in S.

On-demand update. Algorithm 11 outlines the update strategy of PATINA. In order to speed up seeds selection, we propose a strategy that dynamically updates a specific COG-sublist only *when it is demanded*. We refer to this strategy as *on-demand update*. Observe from Algorithm 10 only when a node is selected to be a candidate for S and its marginal gain is not up-to-date with respect to the current S, the update process is invoked for a specific COG-sublist β^r (Line 9 in Algorithm 10). Consequently, a node's marginal gain is not always guaranteed to be valid. Instead, it is updated only when demanded. The algorithm recomputes the marginal gain of $top(\beta^r)$ based on a specific cascade model (Lines 2-4 in Algorithm 11). Observe that we only need to recompute G'_r by random removing edges for R iteration when $v \in V_r$ is

Algorithm 10: The *SeedsSelection* Algorithm.

```
    Input: Graph $G(V,E)$, the budget $k$, the cascade influence function $\sigma(\cdot)$, MAG-list $\mathcal{M}$ and
           COG-sublist $\beta_i$ for $i = 0,\dots,\ell-1$
    Output: Seed set $S$ of nodes, $|S| = k$
1   begin
2       while $\sum_{i=0}^{\ell-1} |S_i| < k$ do
3           $v' = \mathcal{M}[r] = \arg\max_{v \in \mathcal{M}} (v.gain)$;
4           if $v'.valid == 1$ then
5               $S_r.append(v')$;
6               $V.remove(v')$;
7               $V_i.remove(v')$;
8               $\beta^r.remove(v')$;
9           else
10              update($\beta^r$,$G(V_r,E_r)$,$\sigma(\cdot)$) /* Algorithm 11 */;
11          $\mathcal{M}[r] = top(\beta^r)$;
12      return $S = \bigcup_{i=0}^{\ell-1} S_i$;
```

selected. In contrast, state-of-the-art greedy approaches [42] iteratively recompute it over the whole network G for R times after selecting a node into the seed set. That is, it takes $O(Rm)$ operations. Instead, as we have limited the update of the marginal gain to a subnetwork G_r, the time complexity for selecting a node improves to $O(Rm_r)$ (*i.e.,* m_r is the number of edges in the subnetwork G_r).

Next, the algorithm checks whether $top(\beta^r)$ still achieves the highest marginal gain in β^r (Line 5). If it does, then the node's *valid* field is set to 1 (Line 6). Otherwise, it reorders COG-sublist β^r by moving the top element towards the tail to a proper position j such that $\beta^r[j-1].gain > \beta^r[j].gain > \beta^r[j+1].gain$ (Lines 8-12). Observe that our reordering strategy in Lines 5-12 is similar to that of CELF [43]. Finally, the algorithm returns the COG-sublist β^r (Line 13).

For example, consider the aforementioned scenario in Figure 6.2. As discussed earlier, we have selected the first seed v_5. In the next round of seed selection, v_1 is the node with the highest marginal gain. As its gain is not up-to-date, v_1 and β^0 are updated. During the update, the gain of v_1 changes to 0 with respect to the seed $S = \{v_5\}$. Consequently, the algorithm reorders v_1 in β^0 to the tail as it has the least marginal gain.

The aforementioned update strategy makes sense in our partitioning-based influence maximization problem as the gains of elements in a COG-sublist are not affected by other COG-sublists, which results from the fact that a node v is only connected with other nodes in v's COG-sublist. Thus, if v is considered to be selected for the seed set S then it will only affect the marginal gain of those nodes that belong to the same COG-sublist as v. The marginal gain of nodes in other COG-sublists are not affected and need not to be updated. With the aforementioned strategy adopted in Algorithm 8, the memory required for PATINA is only $O(n')$ where n' denotes the number of nodes in the largest component. Hence, it is more efficient than existing algorithms which have $O(n)$ space complexity [4, 42, 43].

Synchronized update. An alternative update strategy, which we refer to as *synchronized update*, guarantees that the nodes in MAG-list are all up-to-date. That is, in this strategy we update all the gains of nodes in β^i whenever an update happens for β^i. Thus, in each iteration $\mathcal{M}[i].valid$ is always guaranteed to be 1 and we can directly select the best node from $\mathcal{M}$ and update the corresponding COG-sublist β^i. For instance, reconsider the aforementioned example. Based on synchronized update strategy, we do not need to wait for checking v_1's *valid* field. Instead, we update β^0 as soon as v_5 is inserted into S, guaranteeing that the nodes in the MAG-list are all valid. Although, this strategy may avoid unnecessary selection of candidate nodes from $\mathcal{M}$, it introduces significant amount of updating and reordering of the COG-sublist. In the next section, we shall experimentally demonstrate the superiority of on-demand update strategy over the synchronized update.

Theorem 6.3 *The time complexity of Algorithm 10 is $O(km'R)$ for Independent cascade model and $O(km'RT)$ for the Weighted cascade model.*

Proof: *(Sketch)* Consider the IC model. Let m_r denote the number of edges in G_r, then every iteration in Lines 3-4 of Algorithm 11 takes $O(m_r)$ time. The reordering of nodes after that

Algorithm 11: The *update* algorithm.

```
Input: COG-sublist $\beta^r = [v_1, v_2, \dots, v_j]$, Subnetwork $G_r(V_r, E_r)$ and the cascade influence
       function $\sigma_r(\cdot)$
Output: Updated COG-sublist $\beta^r$ whose top node's $top(\beta^r).valid = 1$
1  begin
2      for $iter = 1$ to $R$ do
3          compute $G'_r(V_r, E'_r)$ by removing each edge from $G_r(V_r, E_r)$ with probability $1 - p$ (IC
           model) or $1 - 1/v.degree$ (WC model);
4          $top(\beta^r).gain += \sigma_r(top(\beta^r))$
5      if $top(\beta^r).gain \geq \beta^r[1].gain$ then
6          $top(\beta^r).valid = 1$;
7      else
8          foreach $i = 1$ to $j - 1$ do
9              if $\beta^r[i-1].gain < \beta^r[i].gain$ then
10                 $t = \beta^r[i-1]$;
11                 $\beta^r[i-1] = \beta^r[i]$;
12                 $\beta^r[i] = t$;
13     return $\beta^r$;
```

in Lines 8-12 of Algorithm 11 also takes $O(m_r)$ time ($O(m_r T)$ for WC model). Thus, Algorithm 11 takes $O(m_r R)$ time ($O(m_r RT)$ for WC model). According to Algorithm 10, a node's gain should be valid before it is inserted into S. If the node's gain is not valid, Algorithm 11 executes. Hence, Algorithm 10 takes $O(km'R)$ ($O(km'RT)$ for WC model) as m' is the upper bound of m_r. ∎

6.5 PATINA on Distributed Platform

We now discuss how PATINA can be elegantly adopted on a distributed and parallel environment. We utilize the Hadoop [3] framework, which is an open source project maintained by Apache Software foundation and has been widely used in the research of distributed computing [138, 139, 140, 141].

The partitioning of the social network graph results in ℓ non-overlapping subnetworks. As

[3] http://hadoop.apache.org/core/

network	*nodes*	*edges*	*components*	m'
Phy	37,154	231,584	3,883	134,358
Hep	15,233	58,891	1,781	19,630
Wiki-talk	2,394,385	5,021,410	34	5,018,445

Table 6.2: Description of the networks.

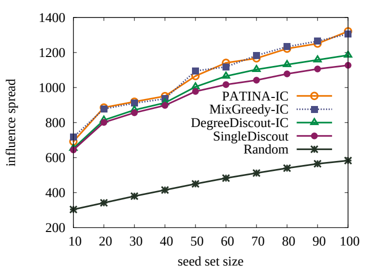
a node in one subnetwork cannot influence nodes in another subnetwork, the marginal gain computation and reordering of nodes in different subnetworks can run in parallel. Ideally, we should distribute ℓ subnetworks to ℓ slave machines. However, due to the limitation of computing resources, we distribute the subnetworks into q slaves ($q < \ell$) each of which contains a set of subnetworks. Furthermore, we store the MAG-list in a master machine, and the COG-sublists are distributed into several slave machines. Each slave machine is in charge of recomputing the marginal gain in one or more subnetworks and outputs the updated top nodes. The master machine manages which COG-sublist the next seed should be selected from. Note that due to our on-demand update strategy, the selection of nodes from MAG-list is independent of the updating and reordering of COG-sublists as we do not need to guarantee each of the COG-sublist is up-to-date. Consequently, selection of nodes from the MAG-list as well as updating and reordering of a COG-sublist can be processed in parallel.

We shall demonstrate that the computation time for influence maximization problem can be significantly reduced by the aforementioned distributed framework in the next section.

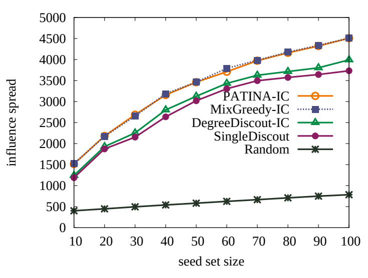
6.6 Performance Study

PATINA is implemented in Java. We run all experiments on 1.86GHz Due-Core Intel 6300 machines with 4GB RAM, running Windows XP. Table 6.2 summarizes the three real-world social network graphs used in our experiments.

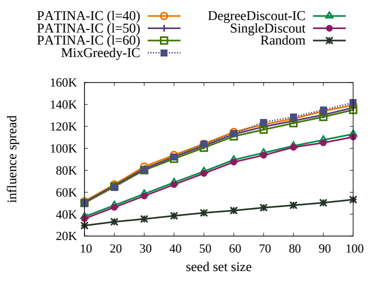
Phy and *Hep* are two academic collaboration networks from the paper lists in two different section of the e-print *arXiv*. Each node in the network represents an author, and the number of edges between a pair of nodes is equal to the number of papers the two authors collaborated.



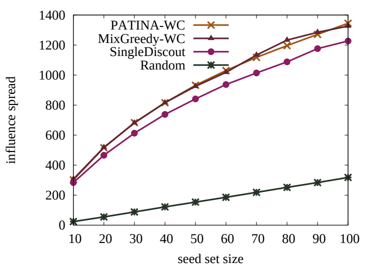
6.3.a: Hep (IC)



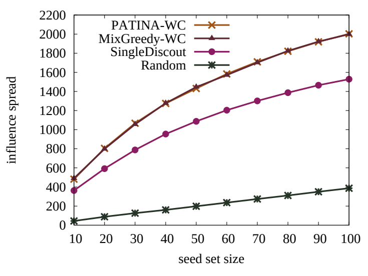
6.3.b: Phy (IC)



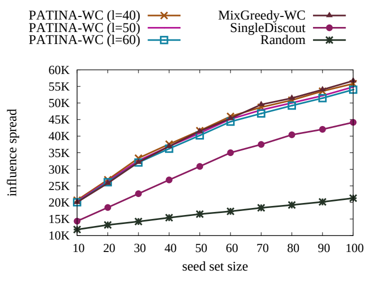
6.3.c: Wiki-talk (IC)



6.3.d: Hep (WC)



6.3.e: Phy (WC)



6.3.f: Wiki-talk (WC)

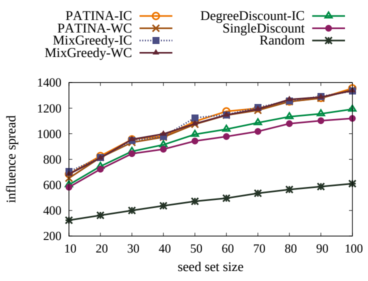
Figure 6.3: Comparison of the number of influenced nodes under IC and WC models.

The *Hep* network is from the "High Energy Physics - Theory" section with papers from 1991 to 2003. The *Phy* network represents the full paper list of the "Physics" section [4] . Note that these datasets are also used in [4, 42]. The *Wiki-talk* [5] is a large network containing millions of nodes representing all the users and discussions in Wikipedia from its inception to January 2008. Nodes in the network represent Wikipedia users and edges represent talk page editing relationship. Note that we consider the edges to be undirected to simplify the discussion.

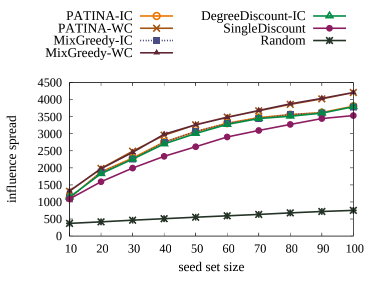
Table 6.2 also specifies the number of components (subnetworks) generated by our BFS partitioning technique and the number of edges in the largest component (m'). Observe that *Phy* and *Hep* are representative networks that can easily be partitioned using BFS technique. On the other hand, *Wiki-talk* is a highly connected large network. The BFS technique failed to

[4]Net and Phy are downloaded from `http://research.microsoft.com/enus/people/weic/graphdata.zip`.

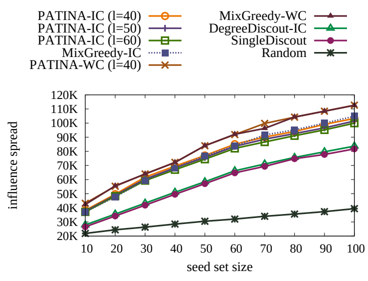
[5]Downloaded from `http://snap.stanford.edu/data/wiki-Talk.html` .



6.4.a: Hep (LT)



6.4.b: Phy (LT)



6.4.c: Wiki-talk (LT)

Figure 6.4: Comparison of the number of influenced nodes under LT model.

generate suitable subnetworks as the largest component in the graph contains over 99.9% of the edges ($m \approx m'$). Hence, we applied the ℓ-way partitioning algorithm provided by CLUTO [136] to generate ℓ subnetworks by removing some edges from the network. In the sequel, we set ℓ to be 40, 50, and 60. The average m' values after partitioning *Wiki-talk* into ℓ subnetworks are 20,452, 16,259 and 15,750 for ℓ equal to 40, 50 and 60, respectively. Specifically, we shall use *Wiki-talk* to study not only the scalability of PATINA but also the quality of seed sets for networks that cannot be partitioned using BFS technique.

We run the following algorithms under both IC and WC models.

- *MixGreedy*-IC*:* The mixed greedy algorithm [42] for the IC model.
- *MixGreedy*-WC*:* The mixed greedy algorithm for the WC model.
- *DegreeDiscount*-IC*:* The degree discount heuristic [42] for the IC model [6] .
- *SingleDiscount:* The single discount heuristic [42] that can be applied to all models.
- PATINA-IC*:* The PATINA algorithm for the IC model.
- PATINA-WC*:* The PATINA algorithm for the WC model.
- D-PATINA-IC*:* The distributed PATINA algorithm for the IC model.
- D-PATINA-WC*:* The distributed PATINA algorithm for the WC model.

[6]Note that we did not select [44] as we were unable to get the implementation from the authors.

- *Random:* As a baseline comparison, simply select k random vertices in the graph. Note that it is also used for experimental study in [4, 42].

We set $T = 5$ (number of iterations in gain computation under WC model) and $R = 20000$ (number of rounds of simulation), which is in line with the experiments in [42]. We vary k from 10 to 100 to evaluate the influence spreads as well as the running times for different seed set size.

6.6.1 Influence Spread

In this experiment, we compare the quality of influence spread (the number of influenced nodes) of representative systems by varying k from 10 to 100. Figure 6.3 reports the influence spreads for different values of k under different cascade models. Note that as the influence spread of distributed PATINA (D-PATINA) is identical to that of PATINA, we only report the results of the latter approach here. We set $p = 0.1$.

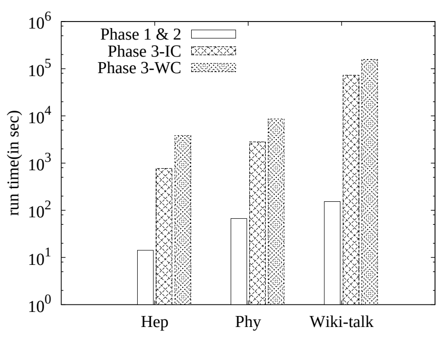
We can make the following observations. Firstly, the PATINA curves follow diminishing pattern which support the submodular nature of influence function. Secondly, the influence spread of the seed sets computed by PATINA is almost identical to those determined by the *MixGreedy* approach for the *Hep* and *Phy* datasets under IC and WC models. We also tested whether PATINA is effective under LT model. We take the seeds selected by PATINA-IC and PATINA-WC and run simulation on both datasets under LT model. The results are depicted in Figures 6.4(a)-(c), which shows that our algorithm is effective under LT model. In summary, our results justifies that our proposed partitioning-based strategy exhibits similar performance to the state-of-the-art greedy approaches when the network can be effectively partitioned using BFS technique.

Figures 6.3(c), 6.3(f), and 6.4(c) depict the influence spread of PATINA on the *Wiki-talk* network for different values of ℓ. Recall that *Wiki-talk* was partitioned using ℓ-way partitioning algorithm which may results in removal of some edges from the network. As ℓ increases the size of each subnetwork may decrease. Consequently, more edges are ignored which may lower the quality of seeds. However, in spite of this, PATINA shows very good performance compared to *MixGreedy*. In particular, PATINA can find seed set that influences at least 90% of the size of influence spread (for $\ell = 40$) computed by *MixGreedy*. In fact, by choosing appropriate value of ℓ (e.g., $\ell = 40$ for *Wiki-talk*), we can generate high quality seed set.

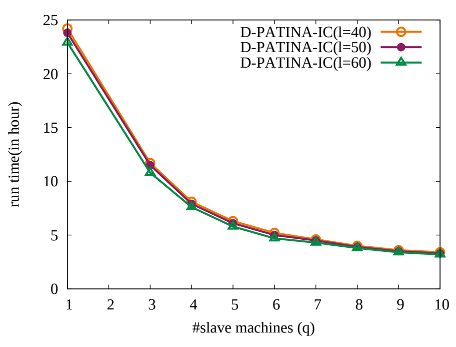
Lastly, both PATINA and *MixGreedy* outperform the two heuristic-based approaches consistently for all networks under all models. Although these heuristics approaches are shown to be orders of magnitude faster than greedy approaches [42], the influence spreads computed by these approaches can be as low as 76% and 60% of the size of influence spread computed by PATINA (or *MixGreedy*) for the *Hep* and *Wiki-talk* datasets, respectively. Both these heuristics approaches discount a node's degree if it has a neighbor selected as a seed. However, discounting the degree does not incorporate the fact that most highest-degree nodes are clustered and hence it cannot avoid unnecessary targeting. Moreover, a node's influence may not always be reflected by its degree in that a node may influence another node over multiple hops while its degree only counts the nodes within a single hop. Importantly, as discussed in Section 6.1, we believe that the seed set quality is paramount to companies as they would like to maximize the influence spreads of their new products. Hence, it cannot be significantly compromised in order to reduce computation time.

6.6.2 Cost of Phases 1 and 2

In this set of experiments, we analyze the cost of Phases 1 (network partitioning) and 2 (MAG-list construction) of PATINA. As discussed in Section 7.3.2, the time complexity of these two steps is $O(m+n)$, which is significantly lesser than the cost of seeds selection. Figure 6.5(a) compares the running times of Phases 1 and 2 (Lines 2-3 in Algorithm 8) with the seeds selection phase (Phase 3) (Line 4 in Algorithm 8) for the three datasets. Observe that the running times of network partitioning and MAG-list construction is significantly smaller than the seeds



6.5.a: Phases 1 and 2



6.5.b: Effect of q

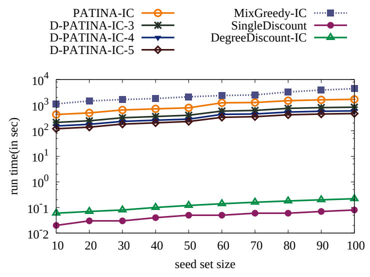
Figure 6.5: Performances of Phases 1 and 2 in PATINA; and affect of q in distributed PATINA.

selection phase, agreeing with our analysis in Section 7.3.2. Note that in order to ensure fair comparison with *Hep* and *Phy*, for *Wiki-talk* we depict only the partitioning time of the ℓ-way partitioning algorithm and not its initial failed attempt to partition using BFS technique.

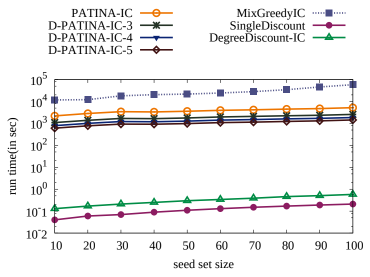
6.6.3 Running Times

We now investigate the response times of various approaches. Although the quality of seed set generated by greedy approaches are consistently superior than those generated by heuristics-based strategies across all cascade models, for the sake of completeness we report the running times of the latter approaches along with the former. Also for the *Wiki-talk* dataset, the response times of PATINA *includes* initial partitioning attempt using the BFS technique.

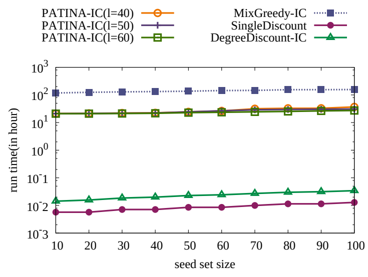
Figures 6.6 and 6.7 report the running times of different approaches on the three datasets based on the IC and WC models. Note that the variable q in "D-PATINA-IC (WC)-q" represents the number of slave nodes in the distributed implementation of PATINA under IC (WC) model. We set $p = 0.1$ and vary k from 10 to 100. We can make the following observations. Firstly, the running times of PATINA and *MixGreedy* increase linearly with the number of seeds k. This justifies our analysis in the preceding sections that the time complexities of both these algorithms are proportional to k. Secondly, PATINA is significantly faster than *MixGreedy* (highest



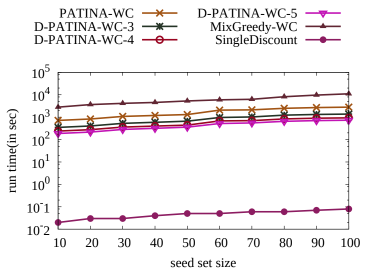
6.6.a: Hep (IC)



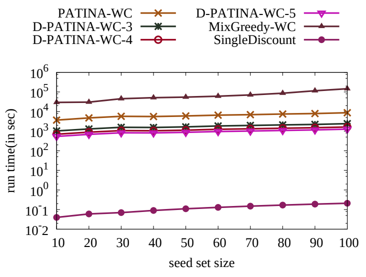
6.6.b: Phy (IC)



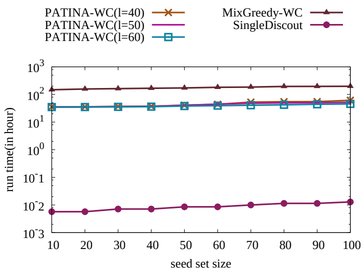
6.6.c: Wiki-talk (IC)



6.6.d: Hep (WC)



6.6.e: Phy (WC)

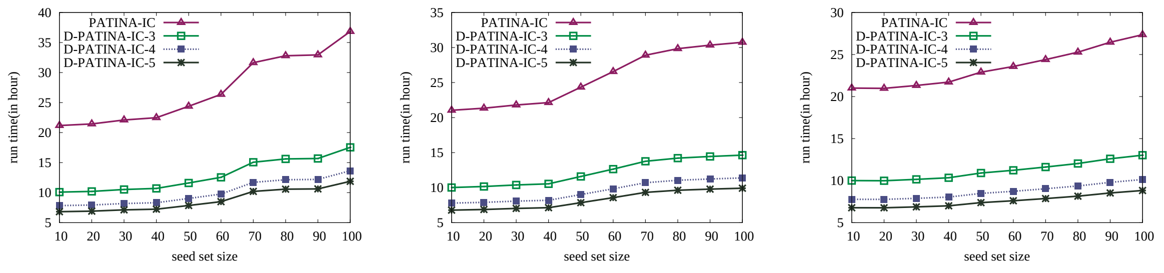


6.6.f: Wiki-talk (WC)

Figure 6.6: Comparison of the running times under IC and WC models.

observed factor being 11) for all datasets and cascade models. Observe that the performance gain of PATINA increases with increase in the size of the network. This is primarily due to the partitioning-based strategy employed by PATINA as well as update and reordering of the MAG-list and COG-sublists. As the size of a network increases, often the increase in size of each subnetwork is not as large as that of the whole network. Instead, the increase in size of a network may often result in the creation of additional subnetworks. Reordering a COG-sublist of a small subnetwork is more efficient than reordering larger subnetworks or entire list of nodes in the network. Consequently, the performance gap between PATINA and *MixGreedy* increases with the size of the network. Thirdly, as mentioned in [42] the heuristic-based techniques are orders of magnitude faster than all greedy algorithms. However, the gain in speed results from sacrificing quality of influence spreads as reported in Section 6.6.1.

Fourthly, distributed implementation of PATINA (D-PATINA) further reduces the cost of



6.7.a: $\ell = 40$ 6.7.b: $\ell = 50$ 6.7.c: $\ell = 60$

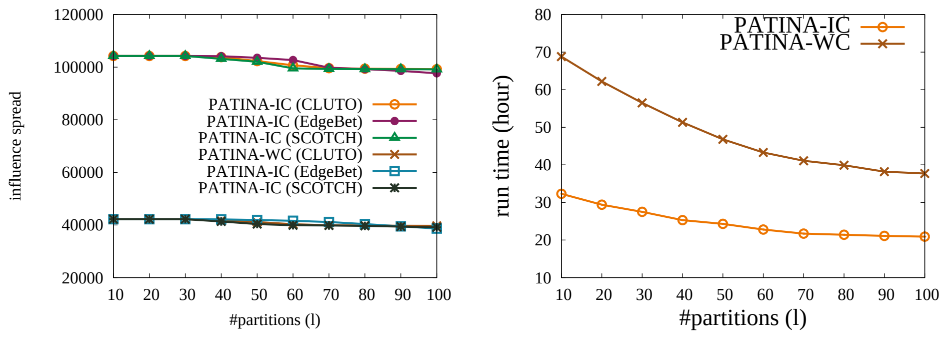
Figure 6.7: Running times in distributed environment for *Wiki-talk* under IC model.

seed set computation. D-PATINA is at least an order of magnitude faster than *MixGreedy* for all datasets. Specifically for the *Wiki-talk* network, *MixGreedy* is prohibitively expensive as it takes more than 4 days to generate the seed set. On the other hand, D-PATINA with just a modest number of slave machines can finish seed set computation in less than 10 hours. This is a reasonable performance on a network with millions of nodes as *a company is often willing to wait for few hours in a day instead of several days to retrieve high quality seed set for its marketing plan*. Lastly, the running times of PATINA and D-PATINA do not change significantly with the variation of number of components (ℓ) for the *Wiki-talk* network.

Next, we test the scalability of (D-PATINA) over *Wiki-talk* by increasing the number of slave machines (q) in Figure 6.5(b). Obviously as q increases the running time decreases sub-linearly indicating that adding more slaves will speedup the algorithm. Particularly, it is less than 6 hours when $q = 10$ whereas *MixGreedy* takes more than 100 hours. In summary, D-PATINA is able to complete the influence maximization task 10-20 times faster than *MixGreedy* while preserving similar result quality. On the other hand, D-PATINA spends a few more hours than the heuristic-based approaches in order to achieve significantly higher quality results.

6.6.4 Evaluation of Partitioning Algorithm

In this part, we compare several partition algorithms towards *Wiki-talk* network and show how sensitive the result is for different partitioning algorithms. We compared three partition al-



6.8.a: Spread 6.8.b: Running time

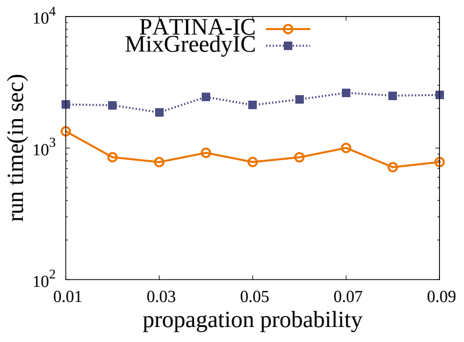
Figure 6.8: Effect of varying the number of partitions ℓ.

gorithms over the experiments on *Wiki-talk* network: CLUTO, *EdgeBetweenness* [129] and SCOTCH [142]. *EdgeBetweenness* method partitions a given graph by removing a specified number of edges that exhibit the highest betweenness score. Both CLUTO and SCOTCH are free graph partitioning packages available online. They are both multi-level algorithms aiming to partition a graph into clusters by removing a limited number of edges. Both CLUTO and SCOTCH partition *Wiki-talk* network around 100 seconds while *EdgeBetweenness* method finishes in more than 20 hours which is almost comparable with the total running time of *Mix-Greedy*.

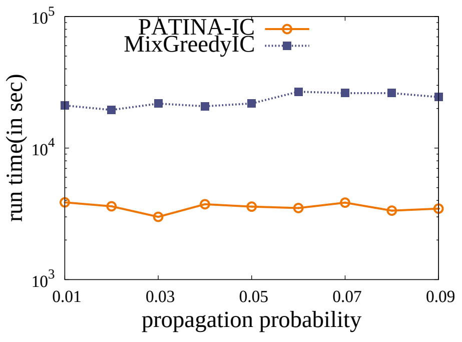
We also tested the quality of influence spread over the result of three partitioning methods. Figure 6.8(a) shows the influence spread result by feeding the partitioned graphs from three algorithms into Algorithm 10. Obviously, the number of influenced node at different value of ℓ do not vary much. Note that, there may exist other partition methods that exhibit more encouraging result than CLUTO or SCOTCH. Actually, PATINA is independent with the detailed partitioning algorithm such that these alternative partitioning methods can also be adopted. In this work, CLUTO is used to partition *Wiki-talk* network if not explicitly stated.

6.6.5 Effect of Propagation Probability

We now study the effect of the propagation probability p under IC model on the running times of the greedy approaches. We vary p from 0.01 to 0.09. Figure 6.9 reports the running times



6.9.a: Hep



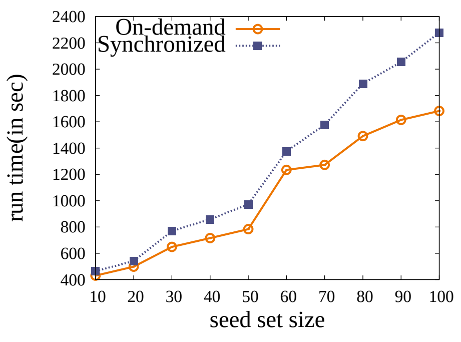
6.9.b: Phy

Figure 6.9: Effect of propagation probability on running times.

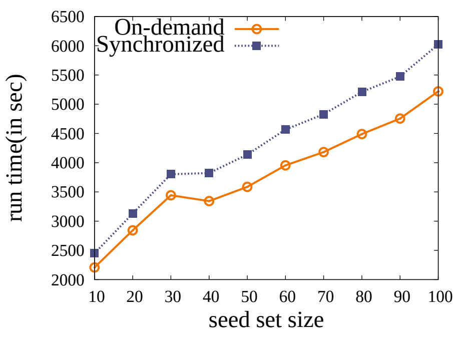
of PATINA-IC and *MixGreedy*-IC on the *Hep* and *Phy* networks. We can make the following observations. First, PATINA-IC is significantly faster than *MixGreedy*-IC for all values of p. Second, the running times of both algorithms do not fluctuate significantly across all p values. As the propagation probability increases, the number of edges after random edge removal process increases as well. However, the time complexities of random removal of edges are $O(m)$ and $O(m')$ for *MixGreedy*-IC [42] and PATINA-IC, respectively. Thus, the propagation probability does not affect the running times significantly. Note that the results are similar for *Wiki-talk* dataset (we omit its discussion here for the sake of space).

6.6.6 Effect of Number of Partitions

We now investigate the effect of number of partitions (ℓ) using the *Wiki-talk* dataset. Figure 6.8 reports the effect of ℓ on influence spread and response time. Note that when ℓ increases the size of each subnetwork may decrease. Consequently, more edges are ignored which may lower the quality of seeds. However, in reality as depicted in Figure 6.8(a) the decrease in the quality of seeds is not significant for both IC and WC models. At the same time, the running time of the models decreases with increase in ℓ.



6.10.a: Hep



6.10.b: Phy

Figure 6.10: On-demand and synchronized strategies.

6.6.7 On-demand vs. Synchronized Update

We compare the on-demand and synchronized update strategies introduced in Section 6.4 and justify our choice of the former. Note that the choice of using one of these strategy only affects the update performance of MAG-list and COG-sublist and not the seed set quality. We set $p = 0.1$ and use the *Hep* and *Phy* datesets under IC model. Figure 6.10 plots the comparison of the running times between the two strategies for different values of k. The running times of both strategies increase linearly with k. Besides, the on-demand strategy is slightly better than the synchronized one which also agrees with our discussion in Section 6.4. The results are similar when we use WC model as well as *Wiki-talk* dataset.

6.7 Summary

The influence maximization problem for online social networks, which focuses on finding the set of k users (seeds) so that they eventually influence the largest number of individuals (influence spread) in the network, is considered important with applications to viral marketing and information dissemination among others. Recently, several greedy and heuristics-based strategies have been proposed to address this problem. Although state-the-art greedy strategies can produce high quality seed set, they are computationally very expensive. For large-scale networks with millions of nodes, the latest greedy approach, *MixGreedy* [42], can take more than four days to find the seed set. On the other hand, recent heuristics-based approaches are orders of magnitude faster but they are susceptible to producing inferior quality seed set. In this chapter, we propose a novel greedy algorithm called PATINA that significantly reduces the computation time without significantly compromising on the seed set quality.

PATINA first partitions the network into a set of non-overlapping subnetworks. Each subnetwork is associated with a COG-sublist which stores the marginal gains of the nodes in the subnetwork in descending order. The node with maximum marginal gain in each COG-sublist is stored in a structure called MAG-list. PATINA exploits these lists along with an on-demand update strategy for marginal gains to efficiently find the seed set. This partitioning-based strategy of PATINA also paves way for its easy adoption on a distributed platform. Specifically, the MAG-list is maintained in a central machine and the maximization of influence for the subnetworks are distributed into several machines and computed in parallel. Our exhaustive empirical study has demonstrated that PATINA and its distributed implementation have excellent real-world performance compared to state-of-the-art greedy approaches.

Seed set quality is of great importance to companies as they would like to maximize the influence spreads of their new products. Kempe et al. [4] was the first to advocate that greedy approximation algorithms produce better quality results compared to heuristics-based techniques. Contrary to this, Chen et al. [42, 44] recently concluded that heuristics-based techniques can produce superior quality results at blazing speed. They envisaged that finding effective heuristics is key to finding scalable solutions to the influence maximization problem. However, they failed to demonstrate that such conclusion related to seed set quality holds for a variety of datasets with different features and different cascade models. In fact, in line with the conclusion of Kempe et al., in this chapter we demonstrated that heuristics-based solutions can produce inferior quality seed set especially for networks like *Wiki-talk*. We believe that partitioning-based greedy strategy on a distributed and parallel framework is a more realistic solution as it can not only improve computation time but also maintain high quality seed set.

Chapter 7

CINEMA: Towards Conformity-Aware Influence Maximization in Large Online Social Networks

In the preceding chapter, we have proposed a novel algorithm called PATINA, which significantly reduces the computation time without compromising on the seed set quality, to solve the *influence maximization* problem in large-scale social networks. PATINA and all the state-of-the-art techniques assume influence propagation follows one of the three cascade models, namely, *independent cascade* model, *weighted cascade* model and *linear threshold* model. In real social-network applications, influence may not simply propagates following these models. For instance, it may also be affected by the influence and conformity of nodes. Hence, in this chapter, we propose a novel cascade model called C^2 (*Conformity-aware Cascade*) model that takes into account the influence and conformity indices of nodes in influence propagation. We propose an algorithm called CINEMA to solve *influence maximization* problem under C^2 model.

The rest of this chapter is organized as follows. We discuss the limitations of existing works and motivate the need for CINEMA in Section 7.1. We propose the C^2 (Conformity-aware Cascade) model in Section 7.2. Next, we solve the influence maximization problem under C^2 by proposing CINEMA algorithm in Section 7.3. We illustrate in detail the modules of CINEMA in Section 7.4 and Section 7.5 in sequence. Experimental results are discussed in Section 7.6. Finally, we conclude this chapter in Section 7.7. Most of the symbols in this chapter are consistent with those used in the preceding chapter. Moreover, similar to Chapter 4, we use $\Phi(\cdot)$ and $\Omega(\cdot)$ to denote the influence and conformity indices of a node, respectively.

7.1 Introduction

With the recent emergence of large-scale online social networking applications (SNA), we are now faced with the opportunity to analyze social network data at unprecedented levels of scale and temporal resolution for marketing, health, politics, communication, education and other applications. However, translating the research techniques of traditional SNA to these large-scale online data-intensive applications is a daunting task. In this chapter, we present our work towards addressing one of the challenges, namely finding a high-quality, real-world, and scalable solution to the *influence maximization* (IM) problem.

Given a social network as well as an *influence propagation* (or *cascade*) model, the problem of *influence maximization* (IM) is to find the set of initial users of size k (referred to as *seeds*) so that they eventually influence the largest number of individuals (referred to as *influence spread*) in the network [4]. For instance, suppose a company wants to market a new product through the social network of potential customers as depicted in Figure 7.1(a). It consists of eight individuals represented by nodes and the edges between them represent connections or relationships between individuals. The company has a limited budget such that it can only select a small number of initial users (significantly less than 8 in Figure 7.1(a)) in the network to use it for free. It hopes that these users will like the product and trigger a cascade of "word-of-mouth" influence by recommending the product to other friends, and many individuals will ultimately adopt it. Figures 7.1(b)-(e) depict the influence spreads for different seed sets. The colored circles denote the nodes that can be influenced by the specified node while dashed circles denote those that are already influenced by current seeds S.

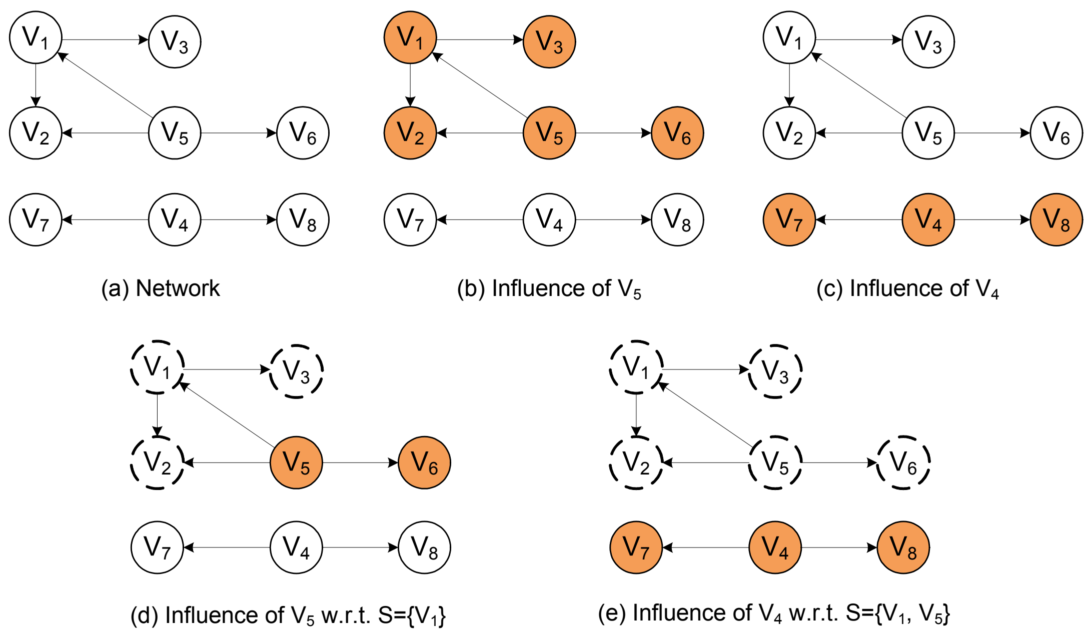

(a) Network
(b) Influence of V5
(c) Influence of V4
(d) Influence of V5 w.r.t. S={V1}
(e) Influence of V4 w.r.t. S={V1, V5}

Figure 7.1: Influence maximization problem.

Domingos and Richardson [64, 66] are the first to study influence maximization as an algorithmic problem. Kempe et al. [4] are the first to consider the problem of choosing the seeds as a discrete optimization problem. They proved that the optimization problem is NP-hard, and presented a greedy approximate algorithm applicable to three cascade models, namely the *independent cascade* (IC) model, the *weighted cascade* (WC) model, and the *linear threshold* (LT) model. A key strength of the proposed algorithm is that it guarantees that the influence spread is within $(1-1/e)$ of the optimal influence spread where e is the base of the natural logarithm. However, deployment of these techniques on large-scale social networks is infeasible as they have poor efficiency and scalability [42]. Recently, several greedy approaches [42, 43] were proposed to address this issue. While these approaches have been able to make significant progress in reducing the computation cost of the IM problem, they still take days to find seeds in real-world networks containing millions of nodes [73]. To alleviate the aforementioned performance bottleneck, several heuristic-based techniques [42, 44, 75, 76] have been proposed which are orders of magnitude faster than the greedy approaches.

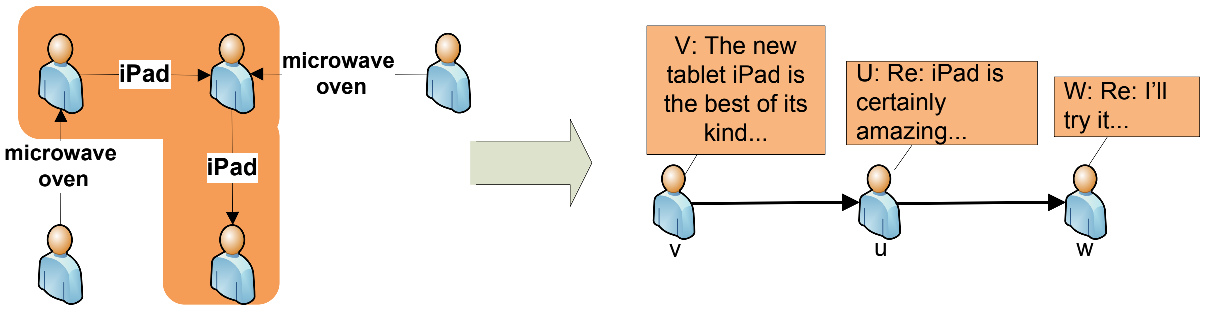

iPad
microwave oven
microwave oven
iPad
V: The new tablet iPad is the best of its kind...
U: Re: iPad is certainly amazing...
W: Re: I'll try it...
v
u
w

Figure 7.2: An example of real-world social network.

The above body of work assumes that a node's ability to influence another is simply determined by an independent probability (*i.e.,* IC) or a probability proportional to the node degree (*i.e.,* WC) or even a binary value controlled by a threshold (*i.e.,* LT). They assume that these probability values are available as input *without addressing the question of how the probabilities are obtained*. Recently, Goyal et al. [73] have shown that such impractical assumptions lead to poor quality seed set and emphasized the need to obtain the influence probabilities from real data. By leveraging on historical *propagation traces* of the underlying network to learn how influence flows in the network, they propose a *credit distribution* (CD) *model* to estimate influence spreads. The authors demonstrate that the CD model ends up choosing seed sets that are very different from the ones chosen by the aforementioned techniques.

7.1.1 Motivation

Consider Figure 7.2 which depicts a fragment of a real-world social network consisting of five individuals. The label of an edge (*e.g.,* "iPad") indicates the topic of conversation between the source and target individuals. To make it more discernible, part of the conversation is magnified in the right hand side. An edge pointing from u to v ($\overrightarrow{uv}$) denotes the influence propagation path with respect to the topic labeled on the edge. Suppose a company wants to present a free trial version of *iPad* to one of these individuals such that she is most likely to recommend her friends to buy *iPad* in future. At first glance, it may seem that this problem can

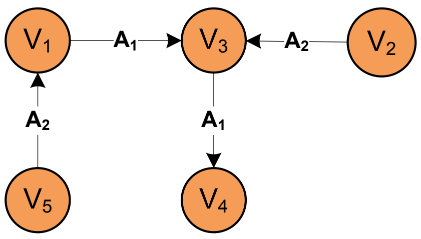


Figure 7.3: Graph representation of Figure 7.2.

Model	$\sigma(v_1)$	$\sigma(v_2)$	$\sigma(v_3)$	$\sigma(v_4)$	$\sigma(v_5)$
IC $(p(\overrightarrow{uv}) = 0.5)$	1.75	1.75	1.5	1	**1.875**
WC $(p(\overrightarrow{uv}) = 1/d(v))$	1.67	1.67	**2**	1	1.83
C^2 $(p(\overrightarrow{uv}) = \Phi_1(u)\Omega_1(v))$	**1.73**	1	1.49	1	1

Table 7.1: Expected influence size of nodes in Figure 7.3.

be solved using an existing influence maximization technique from the above body of work. However, these techniques suffer from the following two limitations that become a stumbling block to the generation of superior quality seed set in real-world networks.

Firstly, the aforementioned IM algorithms are *conformity-unaware*. In real-world networks influence propagation may not simply driven by a node's influence on another but may also depend on the *conformity* of nodes in the propagation path. Specifically, in social psychology *conformity* of an individual refers to a person's inclination to be influenced by others [2]. Hence, *the influence probabilities must be computed by exploiting the interplay of influence and conformity of nodes in the underlying network.*

Let us elaborate on the role of conformity of nodes in influence spread estimation. Reconsider the network in Figure 7.2. We can represent it using the graph depicted in Figure 7.3 where each node denotes an individual. Recall that we aim to select a single seed node ($k = 1$) to propagate a piece of information. Let us review the seed selection in an existing greedy algorithm under IC model first. Assume that influence propagates within the network with probability $p = 0.5$. We need to calculate the expected influence size for all the nodes and select the highest one. Let X be the set of edges that are activated and $\sigma^X(v)$ be the number of nodes that can be reached on activated edge paths from v. Thus, the expected number of

Node ID	$\Phi(\cdot)$	$\Omega(\cdot)$	$\Phi_1(\cdot)$	$\Omega_1(\cdot)$
v_1	0.68	0.21	0.70	0.17
v_2	0.68	0.11	0	0
v_3	0.18	0.94	0.70	0.70
v_4	0.03	0.21	0.17	0.70
v_5	0.18	0.11	0	0

Table 7.2: Nodes' influence and conformity indices.

influenced nodes from v (denoted as $\sigma(v)$) can be expressed as the following [4].

$$\sigma(v) = \sum_X Prob[X] \cdot \sigma^X(v) \qquad \text{(Eq. 7.1)}$$

In the above equation, $Prob[X]$ denotes the probability that all the edges in X are activated. For instance, the expected influence size of v_3 under IC model can be computed as $\sigma(v_3) = Prob[\overrightarrow{v_3v_4} \notin X] \times 1 + Prob[\overrightarrow{v_3v_4} \in X] \times 2$. As $Prob[\overrightarrow{v_3v_4} \notin X]$ or $Prob[\overrightarrow{v_3v_4} \in X]$ equals to 0.5, $\sigma(v_3)$ is 1.5. Table 7.1 reports the expected influence sizes of the five nodes under the IC and WC models (first two rows). Based on Table 7.1 we may select v_5 (resp. v_3) as the seed under IC (resp. WC) model as it exhibits the highest expected influence size. *Unfortunately, this might not be the best choice when conformity of nodes are taken into account.* In real applications the neighbors of v_5 (*i.e.,* v_1) may exhibit different conformity behavior. Observe that v_5 cannot influence anyone else unless $\overrightarrow{v_5v_1}$ is activated. The second and third columns in Table 7.2 report the influence and conformity values of all nodes, respectively, computed using an existing technique [92] (detailed later). Clearly, v_5 exhibits very small influence whereas at the same time v_1 exhibits low conformity. Note that lower the conformity of a node the less likely it is to be influenced by another. In other words, v_1 is not easily influenced by v_5. Consequently, in reality $\overrightarrow{v_5v_1}$ is hardly activated during influence propagation! *Hence, state-of-the-art techniques may generate poor quality seed set as they ignore the conformity of nodes.*

Secondly, existing IM algorithms are *context-unaware*. Observe that the network in Figure 7.2 involves discussions between individuals on two different topics (*i.e., iPad* and *microwave oven*). Specifically, in Figure 7.3 $\overrightarrow{v_1v_3}, \overrightarrow{v_3v_4}$ denote the influence propagation path with respect to topic A_1 (*i.e.,* "`iPad`"); $\overrightarrow{v_5v_1}, \overrightarrow{v_2v_3}$ denote the paths with respect to topic A_2 (*i.e.,* "`microwave oven`"). We refer to such network where topic information are associated with every edge as *context-aware network.* Intuitively, nodes that are involved in the discussion of *iPads* are more likely to purchase *iPads* compared to other nodes. *Unfortunately, existing cascade models are not context-aware as they fail to distinguish between influence propagation related to different topics.* An individual may exhibit different influence and conformity behaviors for different topics. For example, according to the IC model v_5 should be selected as the seed. However, it is clear that v_5 is more interested in *microwave oven* than *iPad*. Hence, v_5 may not recommend *iPad* to her neighbors. Consequently, selecting v_5 as seed is a poor choice as she cannot influence anyone else in the context of topic "`iPad`".

Lastly, existing cascade models assume arbitrarily fixed weights for each edge, such as with the WC. The experimentation and related conclusion about quality of the seed set is doubtful unless the influence strength is learned from real data.

7.1.2 Overview

To address the aforementioned limitations, in this chapter we propose a novel *conformity-aware cascade model* (C^2 model) to study the influence propagation process by taking into account conformity behavior of nodes in a social network. Unlike existing cascade models, the influence probability of a node v in this model is not obtained based on ad hoc assumptions. Rather, we leverage on the *influence* and *conformity values* of v which are computed by analyzing the sentiments expressed by the edges associated with v in the underlying network. We also extend this model to handle context-aware networks, which we refer to as *conformity and context-aware cascade* model (C^3 model). Based on these models, we propose a novel greedy IM algorithm called CINEMA (**C**onformity-aware **IN**flu**E**nce **MA**ximization) that solves the IM problem in both context-aware and context-free social networks.

CINEMA first partitions the network into a set of non-overlapping *components* (subnetworks) and then *distribute* the conformity and context-aware influence maximization computation to these components. A node's influence in one component is not significantly affected by nodes in other components as many large social networks are comprised of series of communities and clusters where a piece of information can easily spread within the community but hard to propagate from one to another [129, 130, 134]. Consequently, each node's influence computation and updates can be limited to the component it resides. Next, for each of these subnetworks, CINEMA computes the *influence* and *conformity indices* of nodes using our recently proposed CASINO algorithm [92]. Note that these indices are used to compute the influence probabilities of nodes. It is worth mentioning that these indices are context-sensitive as each node may exhibit different conformity or influence indices for different topics. Hence, if the network is context-aware then a pair of indices for each node is generated representing its influence and conformity with respect to the specific topic. Otherwise, if the network is context-free, then CASINO will generate a pair of indices for each node indicating its influence and conformity.

Next, CINEMA selects the seed set S from the subnetworks. Specifically, a node v's selection into S is influenced by the conformity indices of the nodes around v at each iteration. A key challenge in this process is to determine the subnetworks from which the seeds need to be selected. To address this issue, we present an efficient data structure called MAG-*list* (**MA**rginal **G**ain List), which stores the *candidate node* having maximum *marginal gain* from each component in the network and guides us to determine the members of seed set. MAG-list is space-efficient as it only requires $O(\ell)$ space complexity, where ℓ is the number of partitioned subnetworks. Thus, in contrast to majority of existing greedy approaches, we do not need to keep the entire collection of nodes of the network in the memory. Additionally, it provides an efficient framework to update the influence of nodes. Note that whenever a node is selected into the seed set, some other nodes' influence may change as well. Thus, it is important to dynamically update the influence of each node. Particularly, CINEMA applies an *on-demand update* strategy in each round to update the MAG-list. Only when a node in the MAG-list is selected as a potential candidate for the seed set, CINEMA updates all the nodes in the *component gain sublist* (COG-sublist) of this node. It is not necessary to update all nodes in the MAG-list.

In summary, the key contributions of this chapter are as follows.

- Firstly, in Section 7.2, we present a novel cascade model called *conformity-aware cascade model* (C^2) which provides a framework to obtain the influence probabilities of nodes from real data by taking into account the conformity of nodes. Further, we extend this model to be context-sensitive.
- Secondly, we present a novel data structure called MAG-*list* which facilitates efficient computation of influence maximization problem (Section 7.4). It also provides an efficient framework to support updates of nodes' influences.
- Thirdly, departing from existing centralized, "non-partitioning-based" solutions to the IM problem, *we take the first step to propose a novel approach that addresses this problem by partitioning the underlying network into a set of non-overlapping subnetworks using an existing network partitioning technique and distributing influence spreads computation to relevant subnetworks* (Section 7.3). Specifically, we present a novel greedy algorithm called CINEMA (Section 7.5) that efficiently exploits the MAG-list build on top of the partitioned subnetworks to compute the seed set for influence maximization under our proposed models while guaranteeing superior quality of the influence spread. Importantly, CINEMA *produces superior quality seed set compared to existing greedy techniques without compromising on the computation cost.* Note that CINEMA is not tightly coupled to any specific graph partitioning technique and as a result its benefits can be realized on any superior graph partitioning approach (in this chapter we chose [136] for reasons justified in Section 7.6).

Symbol	*Definition*
$G(V,E)$	A social network graph
n	number of vertices in G
m	number of edges in G
$G_i(V_i,E_i)$	i^{th} component (subnetwork) in $G(V,E)$
Γ	A set of subnetworks (components)
m'	$\max_{E_i \in E} \|E_i\|$
k	number of seeds to be selected
ℓ	number of connected components (subnetworks)
R	number of rounds of simulation
β^i	A COG-sublist
Υ	A set of COG-sublists
$\mathcal{M}$	MAG-list
S	seed set
S_i	seed nodes selected from $G_i(V_i,E_i)$
$\mathbb{T}$	A topic
$E_{i\mathbb{T}}$	edge correlated with topic $\mathbb{T}$
$G_{i\mathbb{T}}$	subgraph correlated with topic $\mathbb{T}$
$\mathcal{G}$	topic-based subgraph set
$\Omega(\cdot)$	conformity index
$\Phi(\cdot)$	influence index
$\Omega_{\mathbb{T}}(\cdot)$	conformity index with respect to topic $\mathbb{T}$
$\Phi_{\mathbb{T}}(\cdot)$	influence index with respect to topic $\mathbb{T}$
$\overrightarrow{uv}$	the edge pointing from u to v
$\sigma_i(\cdot)$	influence function under cascade model C_i
T	number of iterations in gain computation

Table 7.3: Key notations used in this chapter.

- Lastly, by applying CINEMA to real social networks of various sizes, we show its effectiveness and significant improvement of performance over existing methods (Section 7.6).

The notations used in this chapter are summarized in Table 7.3.

7.2 Conformity and Context-Aware Cascade Models

In this section, we formally introduce a novel cascade model that takes into account conformity of nodes for influence propagation. Next, we extend this model to incorporate topic-related information in context-aware social networks. We begin by briefly describing the classical influence maximization (IM) problem and state-of-the-art cascade models that have been considered in the literature.

7.2.1 Conformity-Aware Cascade Model (C^2 Model)

Recall in last chapter that existing IM techniques that realize the cascade models make ad hoc assumptions regarding the influence probability. Consequently, they do not leverage on several important real-world characteristics (*e.g.,* conformity, topic information) of the underlying data for computing influence probabilities. It is worth mentioning that in [92] we have demonstrated that the presence of an edge between a pair of node u and v is highly affected by the influence of u and the conformity of v. Thus, the probability of influence propagation from u to v is affected by not only influence of u but also conformity of v, which can be computed from the underlying social network (discussed later). Inspired by this finding, we define the *conformity-aware cascade* (C^2) model as follows.

Definition 7.19 *Let A_i be the set of nodes that are influenced in the i-th round and $A_0 = S$. For any $(u,v) \in E$ such that u is already in A_i and v is not yet influenced, v is influenced by u in the next $(i+1)$-th round with a probability that is proportional to the product of u's influence (denoted by $\Phi(u)$) and v's conformity (denoted by $\Omega(v)$). Thus, the probability $v \in A_{i+1}$ can be computed as: $1 - \prod_{u \in A_i, (u,v) \in E} (1 - \Phi(u)\Omega(v))$. This process is repeated until A_{i+1} is empty.*

The C^2 model is suitable for networks where the influence and conformity indices is independent of any specific topic of discussion (*e.g., Epinions*, *Hep*).

7.2.2 Model For Context-Aware Networks

Based on our discussions in the preceding sections, online social networks can be classified into *context-aware* and *context-free* networks. The former represent networks where the edges are associated with topics (context) as social interactions may often involve conversations on specific topics. For example, each conversation in *Twitter* is based on a specific topic. Figure 7.2

depicts interactions between three users on the topic `iPad`. On the other hand, interactions in context-free networks (*e.g., Epinions*) do not involve specific topics. Naturally, in a context-aware network the influence and conformity of nodes are topic-dependent. Thus, we extend C^2 model so that it can be applied to these networks where the influence and conformity indices are correlated with specific topics. We refer to this model as *conformity and context-aware cascade* (C^3) model and is formally defined as follows.

Definition 7.20 *Let A_i be the set of nodes that are influenced by topic $\mathbb{T}$ in the i-th round and $A_0 = S$. For any $(u,v) \in E$ such that u is already in A_i and v is not yet influenced, v is influenced by u in the next $(i+1)$-th round with a probability that is proportional to the product of u's influence (denoted by $\Phi_{\mathbb{T}}(u)$) and v's conformity (denoted by $\Omega_{\mathbb{T}}(v)$). Thus, the probability $v \in A_{i+1}$ can be computed as: $1 - \prod_{u \in A_i,(u,v) \in E} (1 - \Phi_{\mathbb{T}}(u)\Omega_{\mathbb{T}}(v))$. This process is repeated until A_{i+1} is empty.*

Reconsider the example in Section 7.1.1. Recall that v_5 (resp. v_3) is the best node that should be inserted into S under IC (resp. WC) model. However, this may not be true if we take into account the conformity of nodes or topics-related information in the network. The influence and conformity indices of each node is listed in Table 7.2 (fourth and fifth columns). Based on Definition 7.20, $Prob[\overrightarrow{v_3v_4} \notin X]$ can be computed as $1 - (\Phi_1(v_3)\Omega_1(v_4)) = 0.51$. Thus, $\sigma(v_3) = 0.51 \times 1 + 0.49 \times 2 = 1.49$. The expected influences of the remaining nodes with respect to the topic ``iPad'' under C^3 model are listed in Table 7.1 (third row). Thus, we should select v_1 (instead of v_5 or v_3) as the seed when we consider conformity of nodes in a context-aware network. We shall justify our hypothesis empirically in Section 7.6.

Theorem 7.4 *Given a social network graph $G(V,E)$, the influence function $\sigma(\cdot)$ under C^2 (C^3) model is submodular.*

Proof: *(Sketch)* Let S_1 and S_2 be two sets of nodes such that $S_1 \subseteq S_2$. $R(v,X)$ denotes the set of all nodes that can be reached from v on all the activated edges that are in X. Consider the expression of $\sigma^X(S_1 \cup \{v\}) - \sigma^X(S_1)$. It denotes the number of elements in $R(v,X)$ that are not already in $\bigcup_{u \in S_1} R(u,X)$, which is at least as large as the number of elements in $R(v,X)$ that are not in $\bigcup_{u \in S_2} R(u,X)$. That is $\sigma^X(S_1 \cup \{v\}) - \sigma^X(S_1) \geq \sigma^X(S_2 \cup \{v\}) - \sigma^X(S_2)$, which means that the function $\sigma^X(\cdot)$ is submodular. Moreover, we have shown that $\sigma(\cdot)$ can be computed from $\sigma^X(\cdot)$ using Equation Eq. 7.1. It means $\sigma(\cdot)$ is a non-negative linear combination of another submodular function $\sigma^X(\cdot)$. Hence $\sigma(\cdot)$ is also submodular.

7.3 Overview of CINEMA

In this section, we first formally define the partitioning-based influence maximization problem that is proposed in this chapter. Then, we give an overview of key steps of the Algorithm CINEMA.

7.3.1 Partitioning-Based IM Problem

Existing greedy approximation algorithms consume significant time on updating the marginal gains of the top nodes in the list and their rearrangements [4, 42, 43]. Hence, avoidance of unnecessary updates of marginal gains along with reduction of the size of the node list can reduce the computation cost significantly. We achieve this by taking a partitioning-based approach where the whole social network is partitioned into a set of non-overlapping subnetworks. By doing so, we ensure that changes to the marginal gain of a node in a subnetwork G_i do not affect nodes in another subnetwork G_j. Hence, the update of the marginal gains of nodes in G_i is restricted within it instead of the entire network.

Definition 7.21 *Given a budget k and a social network $G(V,E)$, let $\Gamma = \textbf{Partition}(G)$ be the partitions of G containing a set of subnetworks where $V = V_1 \cup V_2 \cup \ldots \cup V_{|\Gamma|}$, $V_i \cap V_j = \varnothing$ $\forall$*

*$i \neq j$, $0 \leq (i,j) < |\Gamma|$, and $(u,v) \notin E$ for $\forall\, u \in V_i, v \in V_j$. Let each G_i exhibits a specific cascade model C_i (*e.g.*, C^2, C^3, IC, WC). Then the **partitioning-based influence maximization problem** finds a set of seeds S in Γ where $|S| = \sum_{i=1}^{|\Gamma|} |S_i| = k$ such that the expected number of nodes that are influenced by S is the largest in G. That is,*

$$S = \underset{\sum |S_i| = k}{\arg\max} \sum_{S_i \subseteq V_i} \sigma(S_i)$$

Observe that in the aforementioned definition we theoretically generalize the problem by adopting different influence models in different subnetworks. Clearly, it can also handle the case where different subnetworks have same cascade model (*e.g.,* C^2 or C^3 model) to reflect many real-world applications. Notably, it is important for the partitioning-based IM problem to guarantee that the influence spread is within $(1 - 1/e)$ of the optimal influence spread.

Theorem 7.5 *Given the social network graph $G(V,E)$ and $\Gamma =$ **Partition**(G), if the influence function $\sigma_i(\cdot)$ for each of the cascade model C_i of $G_i \in \Gamma$ is submodular, then $\sigma(S)$ in Definition 7.21 is also submodular.*

Proof: *(Sketch)* According to Definition 7.21, $\sigma(S)$ can be represented as the following.

$$\sigma(S) = \max_{\sum |S_i| = k} \sum \sigma_i(S_i)$$

Assume $S' \subset S, v \in V_t \setminus S_t$ where $t \in \{1 \ldots \ell\}$ and $S = S_1 \cup S_2 \cup \ldots \cup S_\ell, S' = S'_1 \cup S'_2 \cup \ldots \cup S'_\ell$, then $S'_i \subseteq S_i$. Besides, the following expression holds as $S_i \cap S_j = \varnothing\ \forall\ 0 < (i,j) \leq \ell$.

$$\begin{aligned}
\sigma(S \cup \{v\}) - \sigma(S) &= \sigma_t(S_t \cup \{v\}) - \sigma_t(S_t) \\
\sigma(S' \cup \{v\}) - \sigma(S') &= \sigma_t(S'_t \cup \{v\}) - \sigma_t(S'_t)
\end{aligned}$$

As $S'_t \subseteq S'$ and the influence function $\sigma_t(\cdot)$ is submodular, then $\sigma_t(S_t \cup \{v\}) - \sigma_t(S_t) \leq \sigma_t(S'_t \cup \{v\}) - \sigma_t(S'_t)$ holds according to the definition of submodularity. Thus, $\sigma(S \cup \{v\}) - \sigma(S) \leq \sigma(S' \cup \{v\}) - \sigma(S')$ holds too, which means that the influence function $\sigma(S)$ is submodular.

7.3.2 Algorithm CINEMA

The CINEMA algorithm is outlined in Algorithm 12 and consists of four phases, namely the *network partitioning* phase (Line 2), the *conformity computation phase* (Lines 3-4), the MAG-*list construction* phase (Line 5), and the *seeds selection* phase (Line 6).

Phase 1: The network partitioning phase. For any cascade model, influence always flows along edges in the social network graph. Hence, if there is no path between two nodes then it is not possible for influence to flow between these nodes. In this phase, we first partition the social network graph to a set of non-overlapping connected components (also referred to as subnetworks). As each component is unconnected to another component, the influence computation in a subnetwork is not affected by other subnetworks or components.

Note there are several existing techniques to generate disjoint dense connected components from a graph efficiently [136]. We take the BFS (Breadth First Search)-based strategy to traverse the graph and extract the connected components. The running time of this process is $O(m+n)$. Note that some real-world networks (*e.g., Wiki-talk*) are highly clustered and cannot be easily separated into a set of non-overlapping subnetworks using the BFS technique. Particularly, the BFS-based method may generate components having $m' \approx m$ for these networks. In this case, we partition the network into non-overlapping components using a ℓ-way partitioning algorithm provided by CLUTO (`glaros.dtc.umn.edu/gkhome/cluto/cluto/overview`) [136]. In Section 7.6, we shall justify choosing this graph partitioning algorithm over several existing ones. Given the number of partitions ℓ as input, the algorithm can provide good quality partitions in $O(m)$ time. Note that such partitioning process may inevitably remove some edges in the network. However, as graph partitioning algorithms often minimize the size of edge cuts, the removal of edges does not have significant adverse effect on the estimation of influences of nodes in comparison to existing greedy approaches. Consequently, our experimental results in Section 7.6 demonstrate that for these networks CINEMA can still preserve high quality seed set.

Algorithm 12: The CINEMA algorithm.

```
Input: Graph $G(V,E)$, budget $k$, topic $\mathbb{T}$ (optional) and the cascade influence function
       $\sigma(\cdot)$
Output: Seed set $S$ of nodes, $|S| = k$
1 begin
2     $\Gamma \leftarrow$ NetworkPartition($G$);
3     foreach $G_i \in \Gamma$ do
4         $(G_{i\mathbb{T}}, (\Phi_{i\mathbb{T}}(\cdot), \Omega_{i\mathbb{T}}(\cdot))) \leftarrow$ CASINO($G_i$) /* Based on [92] */;
5     $(\mathcal{M}, \Upsilon) \leftarrow$ MAGConstruction($\Gamma$, $\sigma(\cdot)$, $\Phi_{i\mathbb{T}}(\cdot)$, $\Omega_{i\mathbb{T}}(\cdot)$);
6     $S \leftarrow$ SeedsSelection($G$, $k$, $\sigma(\cdot)$, $\mathcal{M}$, $\Upsilon$, $\Phi_{i\mathbb{T}}(\cdot)$, $\Omega_{i\mathbb{T}}(\cdot)$);
```

In summary, we undertake the following strategy for partitioning the social network graph. If the network can be easily clustered into non-overlapping components by BFS-based method such that $m' \ll m$, then we create the final subnetworks based on this strategy. However, if the BFS-based method fails to generate disjoint components or there exists components after partitioning such that $m' \approx m$, then we adopt the ℓ-way partitioning technique to generate the set of non-overlapping subnetworks.

Phase 2: The conformity computation phase. In this phase, we compute the influence and conformity indices of the nodes in each subnetwork generated from the preceding phase. Note that these indices will be used to compute the influence probabilities based on our C^2 (C^3) model. We invoke the CASINO algorithm [92] for each subnetwork to achieve this goal. For the sake of completeness, here we briefly summarize the algorithm. The reader may refer to [92] for details. Given a subnetwork $G_i(V_i, E_i)$, if it is context-aware then CASINO first extracts a set of subgraphs $\mathcal{G}_i$ where each subgraph $G_{i\mathbb{T}}(V_{i\mathbb{T}}, E_{i\mathbb{T}}) \in \mathcal{G}_i$ contains all the vertices and edges in $G_{i\mathbb{T}}$ associated with a specific topic $\mathbb{T}$. Each subgraph $G_{i\mathbb{T}}$ represents positive or negative attitudes of individuals toward opinions of others in G_i with respect to the topic $\mathbb{T}$. Note that edges of a social network may not be explicitly labeled with positive or negative signs. This is especially true for context-aware networks (*e.g., Twitter*). On the other hand, links in many context-free networks (*e.g., Slashdot*, *Epinions*) are explicitly labeled with signs.

Hence, it is important to assign signs to the edges in each $G_{i\mathbb{T}}$. In CASINO, this is achieved by analyzing the sentiment expressed by the edge. Specifically, for each edge $\overrightarrow{u\mathbb{T}v}$ (the edge pointing from u to v on context topic $\mathbb{T}$) in $G_{i\mathbb{T}}$, it identifies 5-level sentiment (*i.e.,* like, somewhat like, neutral, somewhat dislike, dislike) expressed at both ends using *LingPipe* [99]. If the sentiments at both ends are similar (*sentiment similarity threshold* is less than ε), then the edge is denoted as positive. Otherwise, it is a negative. Hence, each subgraph G_i containing both positive and negative edges can be represented using a pair of graphs $G_i^+(V_i, E_i^+)$ and $G_i^-(V_i, E_i^-)$ denoting the induced graph of positive edges E_i^+ (trust/agreement) and negative edges E_i^- (distrust/disagreement), respectively. Finally, given a set of such signed topic-based subgraphs, CASINO iteratively compute the influence (denoted by $\Phi(.)$) and conformity indices (denoted by $\Omega(.)$) of each individual in each subgraph using the following equations.

$$\Phi(v) = \sum_{\overrightarrow{uv} \in E_i^+} \Omega(u) - \sum_{\overrightarrow{uv} \in E_i^-} \Omega(u) \qquad \text{(Eq. 7.2)}$$

$$\Omega(u) = \sum_{\overrightarrow{uv} \in E_i^+} \Phi(v) - \sum_{\overrightarrow{uv} \in E_i^-} \Phi(v) \qquad \text{(Eq. 7.3)}$$

where $\Omega(u)$ and $\Phi(v)$ represent the conformity and influence indices of nodes u and v, respectively. Observe that a vertex v may have multiple pairs of indices if v is involved in more than one topic-based subgraph. Note that this technique can easily be extended to compute the aggregated indices of an individual by taking into account the entire social network G over all topics.

Phase 3: The MAG-list construction phase. In contrast to the strategy of lazily updating the marginal gains of nodes existing in a single set, in CINEMA the update of marginal gains needs to be carried out within each node set representing each subnetwork *independently*. Given that there may be a large number of subnetworks, how can we efficiently perform the update operations? In this phase, we construct two data structures, namely MAG-*list* and a set of COG-*sublist*s over the subnetworks, that enable us to efficiently determine which subnetwork

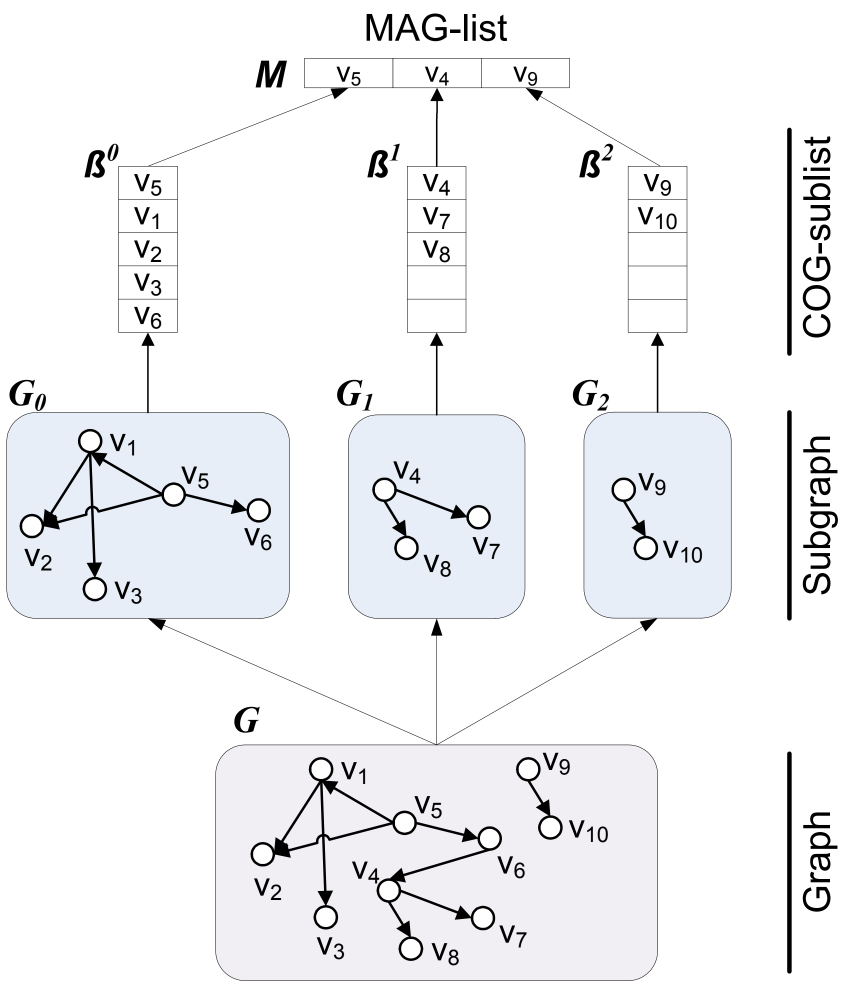


Figure 7.4: The structures of MAG-list and COG-sublists.

the next seed should be selected from and how to effectively perform updates of marginal gains across subnetworks. Informally, a MAG-*list* contains nodes with maximum marginal gain in the subnetworks. Each COG-*sublist i*s associated with a subnetwork or component and stores the marginal gains of all nodes in the subnetwork. We shall elaborate on this phase in Section 7.4.

Phase 4: The seeds selection phase. Lastly, this phase exploits the MAG-list to compute the seed set S from the set of subnetworks (see Section 7.5). It iteratively selects the node having maximum marginal gain from the MAG-list and, if necessary, efficiently updates and reorders nodes in relevant COG-sublists dynamically.

7.4 MAG-List Construction

In this section, we present the MAG-list (**MA**rginal **G**ain List) data structure which we shall be exploiting for the influence maximization problem. For simplicity of discussion, in the sequel we assume the network to be context-free. Extension of our techniques to context-aware networks is trivial. We begin by introducing the notion of ***co**mponent **g**ain sublist* (COG-sublist) which we shall be using to define MAG-list. Given a subnetwork $G_i(V_i, E_i)$ where $G_i \in \Gamma$, the *component gain sublist* of G_i, denoted by β^i, contains the list of nodes V_i. Each node $v \in \beta^i$ and $v \in V_i$ is a 3-tuple $(ID, gain, valid)$ where ID is the unique node identifier of v in G, $gain$ is the marginal gain with respect to S_i, and $valid$ is a boolean variable indicating whether the marginal gain of v is up-to-date. The list is sorted in descending order based on the marginal gains of the nodes. Hence, the node with maximum marginal gain is the top element in the sublist, denoted by $top(\beta^i)$. The *size* of COG-sublist is denoted by $|\beta^i| = |V_i|$. Note that since a social network graph is partitioned into a set of non-overlapping subnetworks, each subnetwork is associated with a COG-sublist.

Informally, a MAG-*list*, denoted by $\mathcal{M}$, contains a list of nodes where each node represents the node with maximum marginal gain in a COG-list. That is, $\mathcal{M}[i] = top(\beta^i)\ \forall\ 0 \leq i < |\Phi|$. Note that the size of $\mathcal{M}$ is the number of non-overlapping subnetworks or components generated from the social network graph G. Figure 7.4 depicts an example of the structures of COG-sublists and MAG-list.

Definition 7.22 *Given the social network graph $G(V,E)$, let $\Gamma = \textbf{Partition}(G)$ where $|\Gamma| = \ell$. Then, the* **MAG-*list***, *denoted by $\mathcal{M}$, is a list of nodes of size ℓ where $\mathcal{M}[i] = top(\beta^i)\ \forall\ 0 \leq i < \ell$.*

To facilitate the discussions on algorithms, we assume some auxiliary functions of nodes. Given a node v, $append(v)$ and $remove(v)$ append and remove v from a node set or COG-sublist, respectively. Algorithm 13 outlines the MAG-list construction algorithm. For each subnetwork

Algorithm 13: The *MAGConstruction* Algorithm.

```
Input: Non-overlapping subnetworks
       $\Gamma = \{G_0(V_0,E_0), G_1(V_1,E_1), \ldots, G_{\ell-1}(V_{\ell-1},E_{\ell-1})\}$ of the social network graph
       $G(V,E)$, the cascade influence function $\sigma(\cdot)$, the influence and conformity
       indices $(\Phi(v), \Omega(v))$ for all $v \in V$.
Output: MAG-list $\mathcal{M}$ and a set of COG-sublists of $\Gamma$ denoted by $\Upsilon$.
1  begin
2      initialize the MAG-list $\mathcal{M}$ of size $\ell$;
3      foreach $G_i(V_i,E_i) \in \Phi$ do
4          initialize COG-sublist $\beta^i$;
5          foreach $v \in V_i$ do
6              $v.valid = 0$;
7              $\beta^i.append(v)$;
8          $\Upsilon.add(\beta^i)$;
9      for $iter = 1$ to $R$ do
10         for $i = 0$ to $\ell - 1$ do
11             compute $G_i'(V_i,E_i')$ by removing each edge $\overrightarrow{uv}$ from $G_i(V_i,E_i)$ with
               probability $1 - \Phi(u)\Omega(v)$;
12             foreach $v \in V_i$ do
13                 $v.gain += \sigma_i(v)$;
14     for $i = 0$ to $\ell - 1$ do
15         sort($\beta^i$) by $\beta^i.gain$ in descending order;
16         $top(\beta^i).valid = 1$;
17         $\mathcal{M}[i] = top(\beta^i)$;
18     return $(\mathcal{M}, \Upsilon)$
```

$G_i(V_i,E_i)$ it first initializes a COG-sublist β_i and populates it by setting the *valid* attributes of the nodes to 0 (Lines 3-8). Next, for nodes in each subnetwork G_i it computes the marginal gains based on the proposed cascade model and assigns them to the list of nodes in β_i (Lines 9-13). The nodes in β_i are sorted in descending order of their marginal gains (Line 15). We set the valid attributes of all $top(\beta^i)$ to 1 as in the first iteration their marginal gains equal to their influences (Line 16). Lastly, the algorithm constructs the MAG-list $\mathcal{M}$ by inserting the top element $top(\beta^i)$ of each β^i (Line 17). Note that the MAG-list construction requires only a linear traversal over the COG-sublists.

7.5 Seeds Selection

Let us first illustrate the seeds selection phase intuitively with the example in Figure 7.4. The MAG-list $\mathcal{M}$ contains the nodes v_5, v_4, and v_9. In the first round of iteration, we select the node having maximum marginal gain from the MAG-list (*i.e.,* v_5) as a candidate. We check if its gain is up-to-date (*valid* field is 1). Recall from Algorithm 13, the top node in each COG-sublist is marked as valid. That is, the marginal gains of all nodes except the top node in a COG-sublist is set to 0 (not up-to-date). Thus, v_5 is valid in this round. Consequently, we insert it into S and remove it from G and the COG-sublist β^0. Now v_1 moves to the top of β^0 and hence it is copied to $\mathcal{M}[0]$. In the next round, assume that v_1 is the node with the maximum marginal gain in $\mathcal{M}$ and hence is selected as a candidate. However, v_1's gain is not up-to-date. Consequently, we need to update v_1's gain as it may change due to addition of v_5 in S. The update process works as follows. We recompute the marginal gain of v_1 in β_0 and check whether v_1's gain is still the highest. If it is, then we mark v_1 as valid. Otherwise, we move v_1 to the correct position in the COG-sublist β_0 to ensure that the list remains sorted in descending order. After the update is completed, we select the next candidate for the next round. The seed selection process terminates when there are k nodes in S.

Algorithm 14 outlines the aforementioned intuition for finding the seed set using the MAG-list. It iteratively selects from the MAG-list the node v' having the maximum gain as a candidate (Line 3). Then the algorithm checks whether the v''s gain is updated by evaluating its *valid* field (Line 4). If it is already updated, then it inserts v' into S_r. Next, it removes v' from the COG-sublist β_r as well as G and continue to the next round (Lines 5-8). Otherwise, the candidate node's gain is not up-to-date. Consequently, the algorithm updates v''s marginal gain and reorders β_r by invoking the *update* procedure (Lines 10), which we shall elaborate later. Then it updates $\mathcal{M}[r]$ using the top element $top(\beta_r)$ (Line 11). The algorithm terminates when there are k nodes in S.

Algorithm 14: The *SeedsSelection* Algorithm.

```
Input: Graph $G(V,E)$, the budget $k$, the cascade influence function $\sigma(\cdot)$, the influence
       and conformity indices $(\Phi(v),\Omega(v))$ for all $v \in V$, MAG-list $\mathcal{M}$ and COG-sublist
       $\beta_i$ for $i = 0,\ldots,\ell-1$
Output: Seed set $S$ of nodes, $|S| = k$
1  begin
2      while $\sum_{i=0}^{\ell-1} |S_i| < k$ do
3          $v' = \mathcal{M}[r] = \arg\max_{v \in \mathcal{M}} (v.gain)$;
4          if $v'.valid == 1$ then
5              $S_r.append(v')$;
6              $V.remove(v')$;
7              $V_i.remove(v')$;
8              $\beta^r.remove(v')$;
9          else
10             update($\beta^r$,$G(V_r,E_r)$,$\sigma(\cdot)$, $\Phi(\cdot)$, $\Omega(\cdot)$) /* Algorithm 15 */;
11         $\mathcal{M}[r] = top(\beta^r)$;
12     return $S = \bigcup_{i=0}^{\ell-1} S_i$;
```

On-demand update. Algorithm 15 outlines the update strategy of CINEMA. In order to speed up seeds selection, we propose a strategy that dynamically updates a specific COG-sublist only *when it is demanded*. We refer to this strategy as *on-demand update*. Observe from Algorithm 14 only when a node is selected to be a candidate for S and its marginal gain is not up-to-date with respect to the current S, the update process is invoked for a specific COG-sublist β^r (Line 9 in Algorithm 14). Consequently, a node's marginal gain is not always guaranteed to be valid. Instead, it is updated only when demanded. The algorithm recomputes the marginal gain of $top(\beta^r)$ based on C^2 (C^3) model (Lines 2-4 in Algorithm 15). Observe that we only need to recompute G'_r by random removing edges for R iteration when $v \in V_r$ is selected. In contrast, state-of-the-art greedy approaches [42] iteratively recompute it over the whole network G for R times after selecting a node into the seed set. That is, it takes $O(Rm)$ operations. Instead, as we have limited the update of the marginal gain to a subnetwork G_r, the time complexity for selecting a node improves to $O(Rm_r)$ (*i.e., m_r* is the number of edges in

network	*nodes*	*edges*	*components*	m'
Phy	37,154	231,584	3,883	134,358
Hep	15,233	58,891	1,781	19,630
Wiki-talk	2,394,385	5,021,410	34	5,018,445

Table 7.4: Description of the context-free networks.

the subnetwork G_r).

Next, it checks whether $top(\beta^r)$ still achieves the highest marginal gain in β^r (Line 5). If it does, then the node's *valid* field is set to 1 (Line 6). Otherwise, it reorders COG-sublist β^r by moving the top element towards the tail to a proper position j such that $\beta^r[j-1].gain > \beta^r[j].gain > \beta^r[j+1].gain$ (Lines 8-12). Observe that our reordering strategy in Lines 5-12 is similar to that of CELF [43]. Finally, the algorithm returns the COG-sublist β^r (Line 13).

For example, consider the aforementioned scenario in Figure 7.4. As discussed earlier, we have selected the first seed v_5. In the next round of seed selection, v_1 is the node with the highest marginal gain. As its gain is not up-to-date, v_1 and β^0 are updated. During the update, the gain of v_1 changes to 0 with respect to the seed $S = \{v_5\}$. Consequently, the algorithm reorders v_1 in β^0 to the tail as it has the least marginal gain.

The aforementioned update strategy makes sense in our partitioning-based IM problem as the gains of elements in a COG-sublist are not affected by other COG-sublists, which results from the fact that a node v is only connected with other nodes in v's COG-sublist. Thus, if v is considered to be selected for the seed set S then it will only affect the marginal gain of those nodes that belong to the same COG-sublist as v. The marginal gain of nodes in other COG-sublists are not affected and need not to be updated. Observe that in CINEMA only the global MAG-list and a *specific* COG-sublist are kept in the memory at an arbitrary timepoint. Hence, the memory required for CINEMA is only $O(n')$ where n' denotes the number of nodes in the largest component. Consequently, it is more efficient than existing algorithms which have $O(n)$ space complexity [4, 42, 43].

Synchronized update. An alternative update strategy, which we refer to as *synchronized update*, guarantees that the nodes in MAG-list are all up-to-date. That is, in this strategy we

Algorithm 15: The *update* Algorithm.

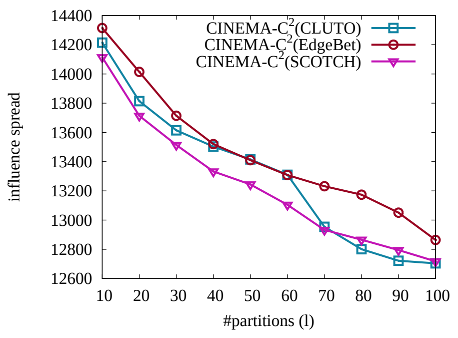
```
Input: COG-sublist $\beta^r = [v_1, v_2, \ldots, v_j]$, Subnetwork $G_r(V_r, E_r)$, the influence and
       conformity indices $(\Phi(v), \Omega(v))$ for all $v \in V$ and the cascade influence function
       $\sigma_r(\cdot)$.
Output: Updated COG-sublist $\beta^r$ whose top node's $top(\beta^r).valid = 1$
1  begin
2      for $iter = 1$ to $R$ do
3          compute $G'_i(V_i, E'_i)$ by removing each edge $\overrightarrow{uv}$ from $G_i(V_i, E_i)$ with probability
           $1 - \Phi(u)\Omega(v)$;
4          $top(\beta^r).gain += \sigma_r(top(\beta^r))$
5      if $top(\beta^r).gain \geq \beta^r[1].gain$ then
6          $top(\beta^r).valid = 1$;
7      else
8          foreach $i = 1$ to $j - 1$ do
9              if $\beta^r[i-1].gain < \beta^r[i].gain$ then
10                 $t = \beta^r[i-1]$;
11                 $\beta^r[i-1] = \beta^r[i]$;
12                 $\beta^r[i] = t$;
13     return $\beta^r$;
```

update all the gains of nodes in β^i whenever an update happens for β^i. Thus, in each iteration $\mathcal{M}[i].valid$ is always guaranteed to be 1 and we can directly select the best node from $\mathcal{M}$ and update the corresponding COG-sublist β^i. For instance, reconsider the aforementioned example. Based on synchronized update strategy, we do not need to wait for checking v_1's *valid* field. Instead, we update β^0 as soon as v_5 is inserted into S, guaranteeing that the nodes in the MAG-list are all valid. Although, this strategy may avoid unnecessary selection of candidate nodes from $\mathcal{M}$, it introduces significant amount of updating and reordering of the COG-sublist. In the next section, we shall experimentally demonstrate the superiority of on-demand update strategy over the synchronized update.

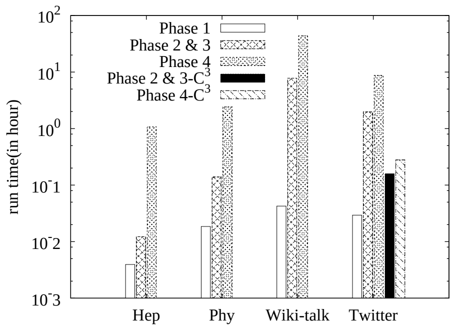
Theorem 7.6 *The time complexity of* CINEMA *is $O(k'm'n' + kTRm')$ where k' is the number of iterations in* CASINO *[92].*

#tweets	*#trends*	*#tweeters*	*#edges*	*#components*	m'
1,054,261	21,917	576,894	1,230,748	24	271,319

Table 7.5: Description of the context-aware *Twitter* network.



7.5.a: Partitioning effect



7.5.b: Phases cost

Figure 7.5: (a) Effect of partitioning algorithm; (b)Cost of different phases.

Proof: *(Sketch)* The time complexity of CINEMA is summation of both the indices computation in CASINO (Line 2 in Algorithm 12) and the influence maximization (Lines 4-6 in Algorithm 12). The time complexity of the former step to compute the indices is $O(k'm'n')$ where k' is the number of iterations in influence and conformity indices computation [92]. On the other hand, the complexity of the latter part is $O(kTRm')$. Hence, the time complexity of CINEMA is $O(k'm'n' + kTRm')$.

7.6 Performance Study

CINEMA is implemented in Java. We run all experiments on 1.86GHz Due-Core Intel 6300 machines with 4GB RAM, running Windows XP. Table 7.5 summarizes the three real-world context-free social network graphs used in our experiments. *Phy* and *Hep* are two academic collaboration networks from the paper lists in two different section of the e-print *arXiv*. Each node in the network represents an author, and the number of edges between a pair of nodes is equal to the number of papers the two authors collaborated. The *Hep* network is from the “High

Energy Physics - Theory" section with papers from 1991 to 2003. The *Phy* network represents the full paper list of the "Physics" section[1]. Note that these datasets are also used in [4, 42, 43, 44, 76]. The *Wiki-talk*[2] is a large network containing millions of nodes representing all the users and discussions in Wikipedia from its inception to January 2008. Nodes in the network represent Wikipedia users and edges represent talk page editing relationship.

We use the *Twitter* dataset to investigate the performance on a context-aware network. The dataset was crawled using the *Twitter* API [3] during Dec 2010 to Feb 2011. We extracted top 20 trends keywords at hourly duration and retrieved up to 1500 tweets for each trend. Then we identified the relationships between all the tweets in the dataset. Table 7.5 reports the statistics associated with this dataset. Note that these statistics are computed after removing non-English tweets (using *Twitter* API). In order to compute accurate influential and conformity indices, we need to have large context-aware network. We removed spam trend keywords which contain only meaningless IDs. Thus, we selected top 492 trends that contain more than 1,000 tweets to compute the indices. For each trend (topic), we identified all the tweets associated to it. Then the edges connecting different tweets using '@' tag are extracted and their signs are assigned as positive or negative. Additionally, there exists another tag 'RT' in many tweets indicating that a tweet author supports another author's opinion by re-tweeting it. That is, if an author u directly re-tweets another twitter v, then it indicates that u wants to distribute this tweet to her followers. Hence, we assign positive signs to such re-tweet edges.

We run the following algorithms under different cascade models.

- *MixGreedy*-IC*:* The mixed greedy algorithm [42] for the IC model.
- *MixGreedy*-WC*:* The mixed greedy algorithm for the WC model.
- *DegreeDiscount*-IC*:* The degree discount heuristic [42] for the IC model.

[1]Net and Phy are downloaded from http://research.microsoft.com/enus/people/weic/graphdata.zip.
[2]Downloaded from http://snap.stanford.edu/data/wiki-Talk.html .
[3]http://dev.twitter.com/doc

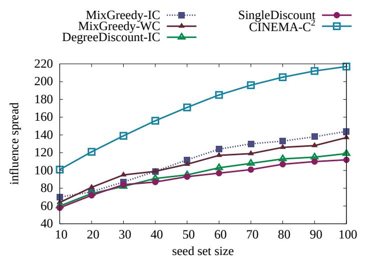
- *SingleDiscount:* The single discount heuristic [42] that can be applied to IC and WC models.
- MIA-N*:* The MIA-N heuristic [77] algorithm that can be applied to IC-N model.
- LDAG*:* The LDAG algorithm [75] for the LT model.
- *SimPath:* The heuristic algorithm [76] for the LT model.
- CINEMA-C^2*:* The CINEMA algorithm for the C^2 model.
- CINEMA-C^3*:* The CINEMA algorithm for the C^3 model.

We set $T = 5$ (number of iterations in gain computation under WC model and C^2/C^3 model) and $R = 20000$ (number of rounds of simulation), which is in line with the experiments in [42]. We vary k from 10 to 100 to evaluate the influence spreads as well as the running times for different seed set size.

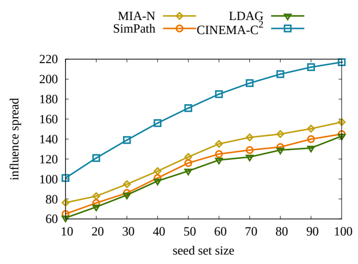
7.6.1 Experimental Results

Effect of Partitioning Algorithms. We first empirically justify the reason for choosing CLUTO as the graph partitioning algorithm for CINEMA. Specifically, we compare three partition algorithms, namely CLUTO, *EdgeBetweenness* [129] and SCOTCH [142], on *Wiki-talk* network and investigate their effects on influence spreads. *EdgeBetweenness* method partitions a given graph by removing a specified number of edges that exhibit the highest betweenness score. Both CLUTO and SCOTCH are multi-level algorithms aiming to partition a graph into clusters by removing a limited number of edges. Both CLUTO and SCOTCH partition *Wiki-talk* in around 100 seconds whereas *EdgeBetweenness* takes more than 20 hours.

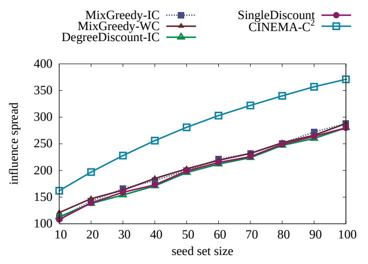
Figure 7.5(a) shows the influence spreads (the number of influenced nodes) generated by feeding the partitioned graphs from these three algorithms into the last three phases in Algorithm 12. Observe that the seeds quality is not affected significantly by these three partitioning methods. Hence, a key advantage of CINEMA is that it is not necessary to be tightly coupled to



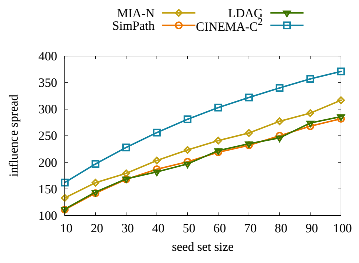
7.6.a: Spread of Hep(a)



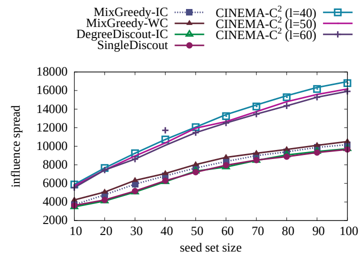
7.6.b: Spread of Hep(b)



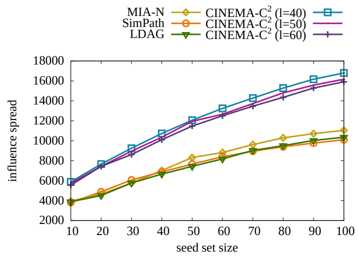
7.6.c: Spread of Phy(a)



7.6.d: Spread of Phy(b)



7.6.e: Spread of Wiki-talk(a)



7.6.f: Spread of Wiki-talk(b)

Figure 7.6: Influence spread results.

any specific partitioning technique. In the sequel, CLUTO is used to partition the networks as it is faster than *EdgeBetweenness*. Note that adoption of a more superior partitioning technique than CLUTO will only enhance the influence spread quality of CINEMA. Also, the increase of ℓ means that more edges are ignored resulting in poorer performance of CINEMA. Note that in practical applications the seed set tends to be small due to budget restriction.

Influence Spread. In this set of experiments, we select k (vary from 10 to 100) nodes using different approaches in the three context-unaware networks and compute the expected influence of those nodes under C^2 model. We set $p = 0.1$ for IC and WC models.

Figures 7.6(a)-(f) report the performances of different approaches. We can make the following observations. Firstly, the CINEMA-C^2 curves follow diminishing pattern which support the submodular nature of influence function. Secondly, it is obvious that existing conformity-unaware algorithms exhibit poor performance with respect to the influence spread result under

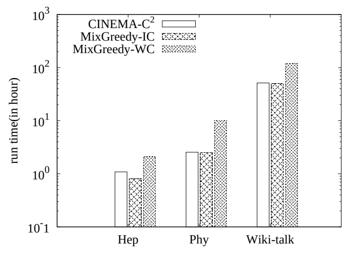
C^2 model. A possible reason for such phenomenon is that these algorithms select seeds greedily under IC and WC models. However, these seeds may not necessarily exhibit expected influence under C^2 model. Hence, unless we take into account the conformity of nodes we cannot produce satisfactory result under C^2 model. Thirdly, Figure 7.6(e,f) depict the influence spread of CINEMA-C^2 on the *Wiki-talk* network for different values of ℓ. Recall that *Wiki-talk* was partitioned using ℓ-way partitioning algorithm which may results in removal of some edges. As ℓ increases the size of each subnetwork may decrease. Consequently, more edges are ignored resulting in slightly lower quality of seeds. In spite of this, CINEMA-C^2 shows superior performance compared to the conformity-unaware techniques. *MixGreedy*-IC and MIA-N exhibit the best performance within existing approaches as shown in Figure 7.6. In fact, influence spread of the results from CINEMA-C^2 outperforms that of *MixGreedy*-IC and MIA-N by about 55% and 41%, respectively.

Lastly, CINEMA-C^2 outperforms the two heuristic-based approaches consistently for all networks. Although these heuristics approaches are shown to be orders of magnitude faster than greedy approaches [42], the influence spreads computed by these approaches can be as low as 42% and 40% of the size of influence spread computed by CINEMA-C^2 for the *Hep* and *Wiki-talk* datasets, respectively. In addition to conformity-unawareness, both these heuristics approaches discount a node's degree if it has a neighbor selected as a seed. However, discounting the degree does not incorporate the fact that most highest-degree nodes are clustered and hence it cannot avoid unnecessary targeting. Moreover, a node's influence may not always be reflected by its degree in that a node may influence another node over multiple hops while its degree only counts the nodes within a single hop. Importantly, as discussed in Section 7.1, we believe that the seed set quality is paramount to companies as they would like to maximize the influence spreads of their new products. Hence, *it cannot be significantly compromised.*

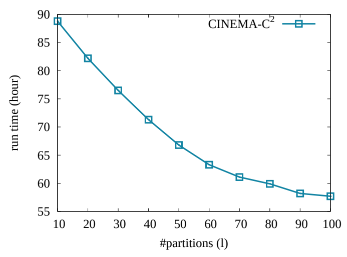
Cost of Phases 1-4. Next, we analyze the cost of Phases 1–4 of CINEMA. Figure 7.5(b) compares the running times of these phases for the four datasets. Since the running time of

CINEMA-C^3				*MixGreedy*-IC	CINEMA-C^2
`Mumford & Sons`	`WeLoveTokioHotel`	`BornThisWayFriday`	`Mubarak`		
3453454	4093419	950596	388397	**8994366**	**8994366**
56068621	191059547	190108655	4725921	**6837510**	950596
3984874	114799747	3498571	5549	143131074	**6837510**
133282617	22418179	147327886	25817119	12732578	4976883
199855121	90276810	49126931	4112233	969858	106789932
8234375	201647517	121158546	957238	**20735827**	179500
4051581	146896900	79897503	3238537	1327826	**20735827**
2894822	97497115	4051581	69290548	2494788	124440574
202658279	204479139	193820280	190736000	**4740643**	**4740643**
780597	206998557	83629945	1314262	1111124	940898

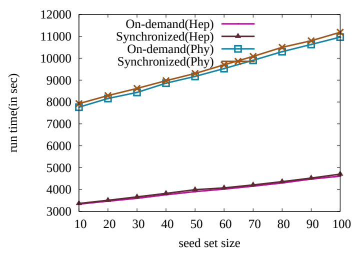
Table 7.6: Seeds selected from Twitter.



7.7.a: Running times



7.7.b: Effect of varying ℓ (Wiki-talk)



7.7.c: On demand vs Synchronized update

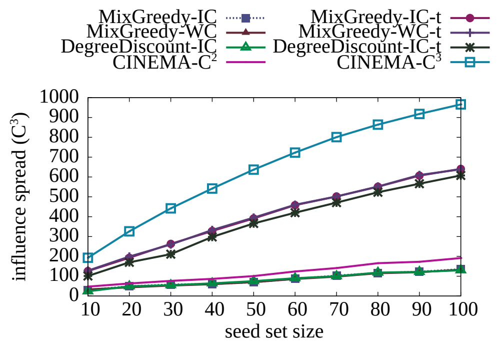
Figure 7.7: (a) Running time comparison; (b)Effect of varying ℓ; (c)On demand update.

MAG-list construction is significantly smaller than the rest, we plot the total running time of Phases 1 and 2. Observe that the seed selection phase dominates the running time agreeing with our analysis in Section 7.3.2. Note that in order to ensure fair comparison with *Hep* and *Phy*, for *Wiki-talk* we depict only the partitioning time of the ℓ-way partitioning algorithm and not its initial failed attempt to partition using BFS technique. For *Twitter* we compared the running times in both C^2 and C^3 models. As the first phase in both models are identical, there are two additional bars representing Phases 2 and 3, and Phase 4 in C^3 model, respectively.

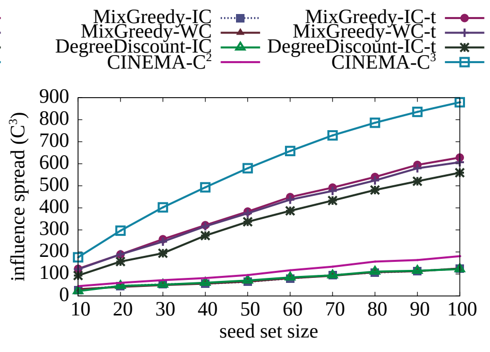
Running Times. We now investigate the response times of various approaches. For the *Wiki-talk* dataset, the response times of CINEMA-C^2 *includes* initial partitioning attempt using the BFS technique. Figure 7.7(a) reports the running times of different approaches. Observe that in spite of the additional steps of network partitioning and indices computation, the running times of CINEMA-C^2 is almost the same with *MixGreedy-IC* and much less than *MixGreedy-WC*. Thus, it is reasonable to be applied in real applications. Since it is already demonstrated in [42] that the heuristic-based techniques are orders of magnitude faster than all greedy algorithms, for the sake of visual clarity, we do not plot them here. However, the gain in speed results from sacrificing quality of influence spreads as reported above. Lastly, we study the effect of varying ℓ results on the running time of CINEMA-C^2. Figure 7.7(b) depicts that the running time of CINEMA-C^2 over *Wiki-talk* network. Observe that the running time decreases as ℓ increases.

On-demand vs. Synchronized Update. Lastly, we compare the on-demand and synchronized update strategies introduced earlier and justify our choice of the former. Note that the choice of using one of these strategy only affects the update performance of MAG-list and COG-sublist and not the seed set quality. Figure 7.7(c) plots the comparison of the running times between the two strategies for different values of k for the *Hep* and *Phy* datasets. The running times of both strategies increase linearly with k. Besides, the on-demand strategy is slightly better than the synchronized one which also agrees with our discussion in the preceding section. The results are similar when we use the *Wiki-talk* and *Twitter* datasets. Due to space constraints, we do not present it here.

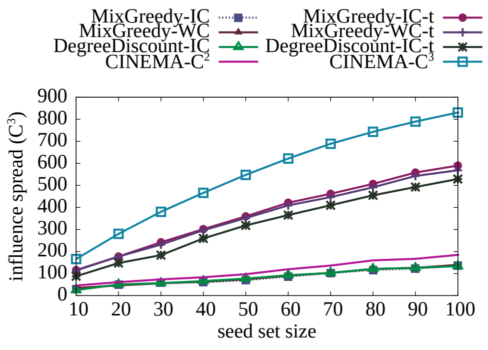
CINEMA on Context-aware Network. In order to study the influence spread in context-aware network, we perform a set of experiments on the *Twitter* dataset. Recall that if the network is context-aware then we compute the influence and conformity indices of nodes with respect to the given topic $\mathbb{T}$ in each partitioned subgraphs. Thus, a pair of topic-specific influence and conformity indices is associated with each node. Then we can maximize the influence with respect to a given topic $\mathbb{T}$ by selecting k seeds using Algorithm 12. We also create context-aware variants of *MixGreedy* (denoted by *MixGreedy*-IC/WC-t), *SimPath*, LDAG, MIA-N and



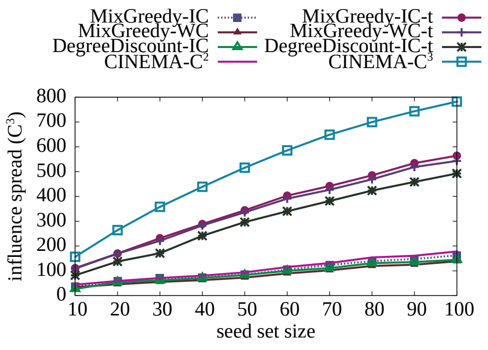
7.8.a: `Mumford & Sons`



7.8.b: `BornThisWayFriday`



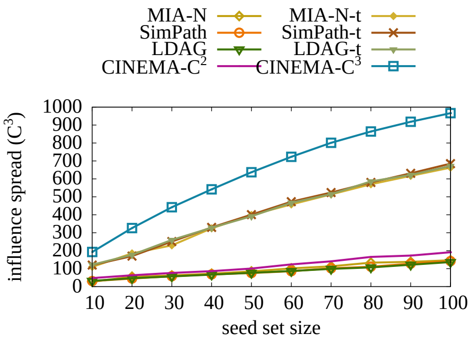
7.8.c: `WeLoveTokioHotel`



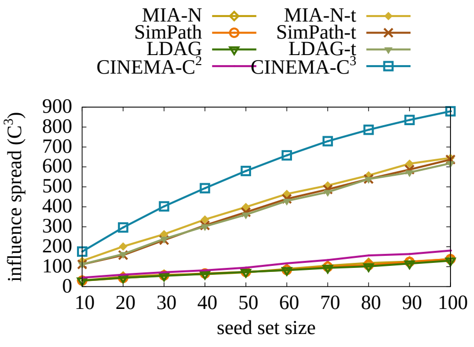
7.8.d: `Mubarak`

Figure 7.8: Influence spread on a context-aware network. (Comparison with *MixGreedy* and *DegreeDiscount*-IC)

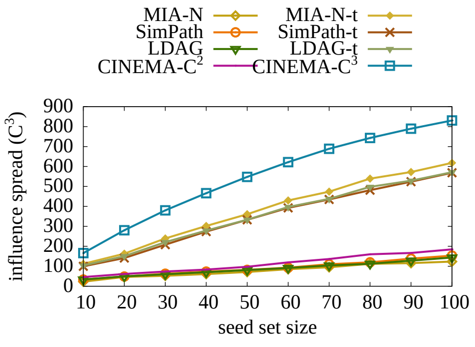
DegreeDiscount (denoted by *DegreeDiscount*-IC-t) where the topic-specific subgraph related to topic $\mathbb{T}$ is first extracted from the social network graph and then *MixGreedy* (resp. *SimPath*, *DegreeDiscount*, MIA-N) is executed on the subgraph. Figures 7.8 and 7.9 plot the influence spreads for the top 4 topics with the maximum number of tweets (`Mumford & Sons` , `BornThisWayFriday` , `WeLoveTokioHotel` and `Mubarak`). Clearly, CINEMA-C^3 consistently outperforms all existing methods as well as CINEMA-C^2. This is because the context-unaware approaches are unable to distinguish whether a seed exhibits any influence with respect to a specific topic $\mathbb{T}$. Many of the seeds selected by these approaches may not influence anyone else for topic $\mathbb{T}$. Thus, the influence spreads from these seeds are poor. Notably CINEMA-C^3 is not only *context-aware* but also *conformity-aware*. Hence, it significantly outperforms



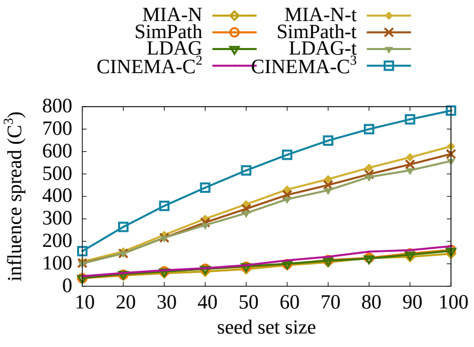
7.9.a: `Mumford & Sons`



7.9.b: `BornThisWayFriday`



7.9.c: `WeLoveTokioHotel`

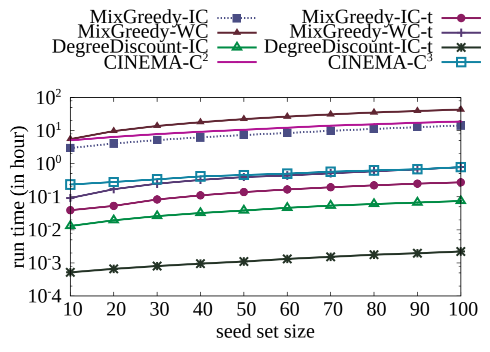


7.9.d: `Mubarak`

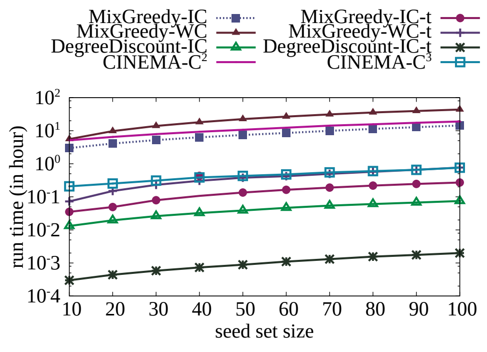
Figure 7.9: Influence spread on a context-aware network. (Comparison with MIA-N, *SimPath* and LDAG)

MixGreedy-IC/WC-t, MIA-N-t and *SimPath*/LDAG-t.

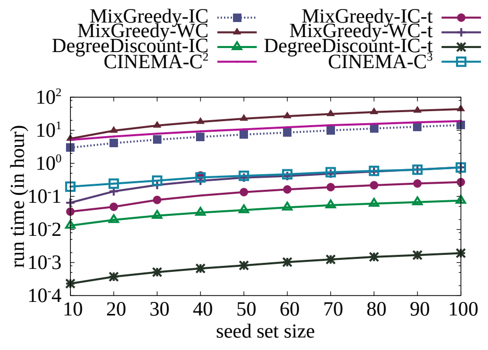
Table 7.6 reports *Twitter IDs* of seeds selected by different models for $k = 10$. The first four columns report the seeds computed by CINEMA-C^3 for four different topics. The seeds set computed by *MixGreedy*-IC and CINEMA-C^2 over entire network are shown in the last two columns. We can make two key observations. Firstly, the seeds identified by CINEMA-C^3 are completely distinct from those selected by context-unaware techniques (*MixGreedy* and CINEMA-C^2). In other words, it further strengthen our conclusion that the result quality can improve significantly if topic information is incorporated in the IM problem. Secondly, the seeds generated by CINEMA-C^2 and *MixGreedy*-IC are significantly different (only 4 out of 10 seeds appear in both sets) highlighting the importance of conformity-awareness for IM problem.



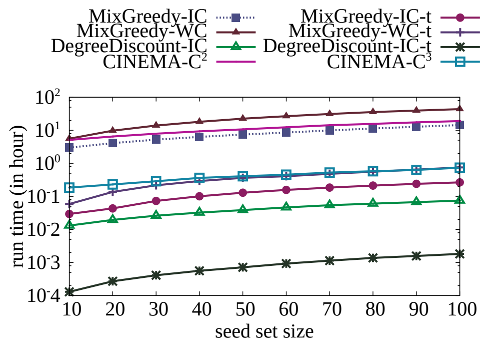
7.10.a: `Mumford & Sons`



7.10.b: `BornThisWayFriday`



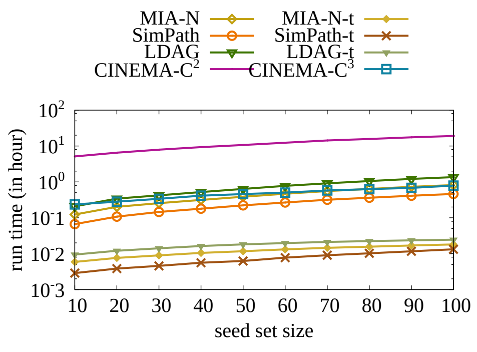
7.10.c: `WeLoveTokioHotel`



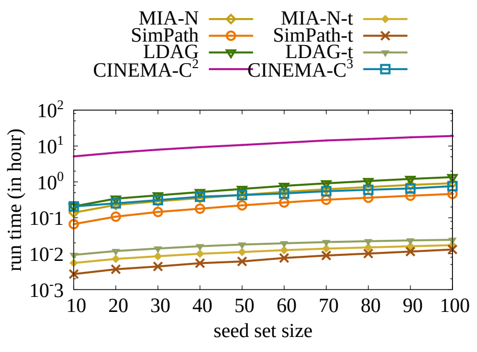
7.10.d: `Mubarak`

Figure 7.10: Running times on a context-aware network. (Comparison with *MixGreedy* and *DegreeDiscount*-IC)

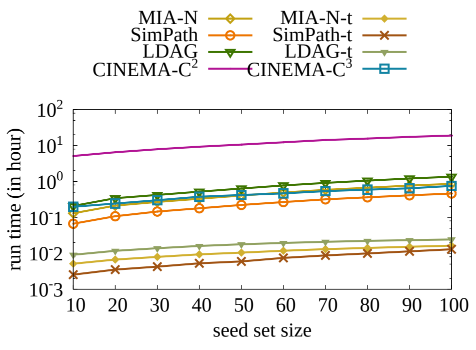
Finally, we compare the running times of different approaches in context-aware network. Figures 7.10 and 7.11 plot the running times of the benchmark techniques for the top 4 topics. Observe that the context-aware approaches (*i.e.,* CINEMA-C^3, *MixGreedy*-IC/WC-t, *SimPath*/LDAG-t, MIA-N-t and *DegreeDiscount*-IC-t) spend less than an hour to select seeds for each topic. In contrast, the context-unaware greedy approaches (*i.e., MixGreedy*-IC/WC and CINEMA-C^2) spend 10 hours to select seeds. This is because context-aware approaches only select seeds from the nodes which are related to the given topic and thus significantly reduce the network size. Most importantly, *conformity and context-aware* CINEMA *not only produces superior quality influence spread but also takes reasonable time to compute it, making it viable for real-world applications*.



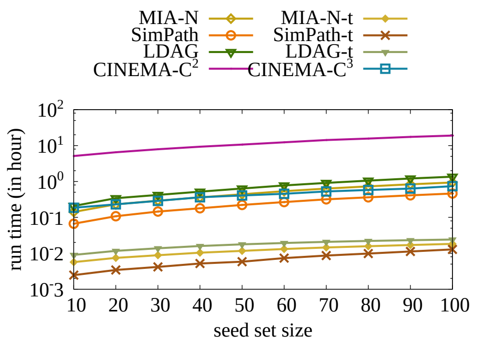
7.11.a: `Mumford & Sons`



7.11.b: `BornThisWayFriday`



7.11.c: `WeLoveTokioHotel`



7.11.d: `Mubarak`

Figure 7.11: Running times on a context-aware network. (Comparison with MIA-N, *SimPath* and LDAG)

7.7 Summary

State-of-the-art greedy strategies of influence maximization algorithms claimed that they can produce high quality seed set. However, in this chapter we showed that this may not be true for real-world applications. This is because all existing algorithms have ignored two important factors in social influence propagation, namely, *conformity* and *topic*. Nodes may exhibit different inclination on conforming to others, thus influence propagation also depends on conformity. Moreover, the conformity is topic-aware that a node may exhibit different conformity with respect to different topics. In order to formally model the influence propagation process with respect to conformity and specific topic, we propose a novel cascade model called

Conformity-aware Cascade model. According to the model, influence propagation from u to v depends on not only $\Phi_A(u)$ but also $\Omega_A(v)$ with respect to the given topic A. Based on the model, we propose a novel algorithm called CINEMA-C^3 that takes into account the effect of conformity and topic information in influence propagation. Moreover, CINEMA-C^3 can easily find seeds with respect to a specific topic. Experimental results show that the results generated from CINEMA-C^3 are significantly better than existing algorithms.

Chapter 8

Conclusions and Future Work

In this chapter, we summarize the contributions of our work and propose several future research directions.

8.1 Summary

In this thesis, we have proposed a framework called *i*MASON (influence-driven Multi-level Analysis of SOcial Networks). Within this framework, we developed several models focusing on mining social influence effect within large online social networks.

The contributions of this thesis can be summarized as the follows.

- Information propagation within the blogosphere is of much importance in implementing policies, marketing research, launching new products, and other applications. In this thesis, we take a microscopic view of the information propagation pattern in blogosphere by investigating blog cascade affinity (Chapter 3). A blog cascade is a group of posts linked together discussing about the same topic, and cascade affinity refers to the phenomenon of a blog's inclination to join a specific cascade. We identify and analyze an array of *macroscopic* and *microscopic* content-oblivious features that may affect a blogger's cascade joining behavior and utilize these features to predict cascade affinity of blogs. Based on these features, we present non-probabilistic and probabilistic strategies, namely SVM classification-based approach and *Bipartite Markov Random Field*-based (BiMRF) approach, respectively, to predict the probability of blogs' affinity to a cascade and rank them accordingly. Evaluated on a real dataset consisting of 873,496 posts, our experimental results demonstrate that our prediction strategy can generate high quality results ($F1$-measure of 72.5% for SVM and 71.1% for BiMRF). Our experiments also show that among all features identified, the *number of friends* is the most important factor affecting bloggers' inclination to join cascades.

- In this thesis, we propose a novel algorithm CASINO (**C**onformity-**A**ware **S**ocial **IN**fluence c**O**mputation) to study the interplay between influence and conformity of each individual (Chapter 4). Given a social network, CASINO first extracts a set of topic-based subgraphs where each subgraph depicts the social interactions associated with a specific topic. Then it optionally labels the edges (relationships) between individuals with positive or negative signs. Finally, it computes the influence and conformity indices of each individual in each signed topic-based subgraph. Our exhaustive empirical study with several real-world social networks demonstrates superior effectiveness and accuracy of CASINO compared to state-of-the-art methods. Furthermore, we reveal several interesting characteristics of "influentials" and "conformers" in these networks.

- People in large online social rating networks (*e.g., Epinions*, *Blippr*) form product communities. A potential member can join a product community by giving a new rating to the product. We refer to this phenomenon of a product community's ability to "attract" new members as product affinity. The knowledge of a ranked list of products based on product affinity is of much importance to be utilized for implementing policies, marketing research, online advertisement, and other applications. In this thesis, we identify and analyze an array of features that exert effect on product affinity and propose a novel model, called *AffRank*, that utilizes these features to predict the future rank of products according to their affinities (Chapter 5). Evaluated on two real-world datasets, we demonstrate the effectiveness and superior prediction quality of *AffRank* compared to baseline methods. Our experiments show that features such as *affinity rank history*, *affinity evolution distance*, and *average rating* are the most important factors affecting future rank of products. At the same time, interestingly, traditional community features (*e.g.,* community size, member connectivity, and social context) have negligible influence on product affinities.

- Influence maximization is the problem of finding a small subset of nodes (seed nodes) in a social network that could maximize the spread of influence. Since this optimization problem is NP-hard, recently several *greedy* and *heuristics*-based strategies have been proposed to address it. Although state-of-the-art greedy strategies can produce high quality seed set, they are computationally very expensive especially for large-scale networks. On the other hand, *heuristics*-based approaches are orders of magnitude faster but they often produce inferior quality seed set. In this thesis, we propose a novel greedy algorithm called PATINA that significantly reduces the seed set computation time while maintaining superior seed set quality (Chapter 6). It first partitions the network into a set of non-overlapping subnetworks. Each subnetwork is associated with a COG-*sublist* which stores the marginal gains of the nodes in the subnetwork in descending order. The node with maximum marginal gain in each COG-sublist is stored in a data structure called MAG-*list*. These structures are manipulated by PATINA to efficiently find the seed set. A key feature of PATINA is that each node's influence computation and updates can be limited to the subnetwork it resides instead of the entire network. Our partitioning-based approach also ensures that PATINA can be easily adopted on a distributed platform. Our exhaustive empirical study with large-scale real-world social networks demonstrate that PATINA and its distributed variant have superior real-world performance compared to a latest state-of-the-art greedy approach.

- We propose a novel algorithm called CINEMA which takes into account the conformity of nodes to solve influence maximization problem (Chapter 7). Influence propagation may not simply depend on a probability (*i.e.,* IC model) or the degree of node (*i.e.,* WC model), it may also depend on the influence and conformity at both ends of the edge. Moreover, state-of-the-art techniques are not suitable for influence maximization over a specific topic. Thus, we propose a novel Conformity-aware Cascade model (C^2) which incorporates conformity of nodes and topic information in influence propagation. We propose a novel algorithm called CINEMA which can produce satisfactory result under C^2 model. Moreover, experimental results show that CINEMA not only produce significantly better result than state-of-the-art technique but also can be applied in topic-conscious influence maximization task.

8.2 Future Research Directions

We believe our work in this thesis opens up several important directions for future research work on social influence study, including all the three levels of targets: individual, community and network. We briefly discuss some open questions which have not been addressed yet. We believe these questions are important in social influence mining and will be our focus in future work.

8.2.1 Individual-level

In Chapter 3, we have proposed a series of novel features to predict the blog cascade affinity. In the prediction we assume that the features proposed are independent with each other. However, this may not be true. For example, the feature *number of friends* may exhibit some correlation with *number of participants*. As part of future work, we intend to investigate the correlation between different features and how they influence the cascade affinity. Moreover, a future research goal in this field is to investigate the exact pattern of human affiliation behavior (*i.e.,* *How do people join community/buy product*?) in social networks. All the existing works in this field only study a limited set of features and test whether they exhibit effect on human behavior. However, human affiliation behaviors in social networks may be affected by very complex pattern that is hard to be described simply using a series of features. Finding the exact pattern of human affiliation behavior in social network is challenging and requires careful investigation.

In Chapter 4, we have proposed CASINO which evaluate the influence and conformity of individual nodes in an arbitrary social network. As a part of our future work, we are also interested in investigating the role of the two causes of conformity, namely, influence and normative, to the influence analysis problem. Aside from *conformity*, there are a series of other concepts proposed in social psychology research which have not been tested in real-world online social networks. For example, *social facilitation* is the tendency for people to do better on simple tasks when in the presence of other people [143]. To the best of our knowledge, the idea has not been tested in online social networks. A future research goal of our work includes investigating ways of testing this phenomenon and utilizing it in real applications.

8.2.2 Community-level

As part of our future work with respect to community-level social influence study, we intend to investigate prediction of future ranks of *newly-released* products (products that appear in the social rating networks less than a month ago). These types of products do not have sufficient historical information in the SRNs to make accurate prediction. Hence, *external* information (*e.g.,* online news media) may need to be exploited for predicting product affinities. Besides, a challenging task extending the research in this field lies in how to utilize these features to design online marketing strategy such that a new product can receive more positive ratings. More advanced techniques need to be explored in order to improve the overall ratings of a product. For example, there exists another social psychology concept called *group polarization* which refers to the tendency for groups to make decisions that are more extreme than the initial inclination of its members [144]. However, it has not been tested in real-world online social rating networks. A future goal in the research of these product communities in social rating networks lies in investigating ways of utilizing a series of phenomenons in social psychology (*i.e., group polarization*) to improve the social ratings of a product. A carefully designed application with respect to the phenomenon may provide potential benefit to viral marketers.

8.2.3 Network-level

In Chapter 6, we have proposed PATINA to solve influence maximization problem in both single node mode and parallel mode. As for future work, we plan to explore more efficient parallel solution to the problem. Besides, the current problem setting is based on several popular submodular cascade models. It is possible for cascade models of real-world networks to be far more complex and they may not exhibit submodular characteristics. Hence, we plan to investigate strategies to efficiently solve the partitioning-based influence maximization problem for these complex networks.

In Chapter 7 we have proposed a novel cascade model C^2. According to the model, influence propagates through an edge $\overrightarrow{uv}$ with a probability proportional to influence of u and conformity of v. However, influence propagation in real-world networks may follow much more complex patterns. Further investigation over the cascade models in real-world networks is required. A long-term research goal in the field of influence maximization lies in the application of these algorithms. Whether applying these algorithms in real-world applications can increase the benefit of online advertisers has not been tested yet. It will be of much importance for online marketers if these algorithms can be applied in real-world scenarios.

On the other hand, conformity study in social network analysis is in an early stage, other varieties of conformity evaluation model and cascade model may be developed afterwards. There exist many future extensions in CASINO and C^2 model. For example, conformity and influence indices may be learned from the network instead of computed from links, C^2 model may incorporate other factors aside from influence and conformity.

8.2.4 The Correlation Between the Social Influence Effect at Different Levels

As discussed in the first chapter, we analyze the social influence phenomenon at network-level, community-level and individual-level. However, these are not sufficient to understand the evolution in social networks. The social influence effect in different levels may exhibit impact on each other. To the best of our knowledge, the correlation between these different levels has not been addressed yet. Moreover, a future goal of our research work is to design a uniform implementation incorporating all our findings of social influence in three levels as well as the correlation between different levels. It is challenging to make this implementation easily applied to arbitrary social networks and automatically extract informative social influence effect at different levels such that a complete view of social influence in the network is unveiled.

8.3 Conclusion

In this thesis, we have presented a framework called *i*MASON to investigate the social influence effect in real-world online social networks. Our research is conducted over a series of real-world datasets and spans several research areas such as machine learning, information retrieval and social psychology. It will allow us to tackle more complex social influence problems in future. In summary, our work in this thesis has opened a series of future research works.

References

[1] J. Leskovec, M. McGlohon, C. Faloutsos, N. Glance, and M. Hurst, “Cascading behavior in large blog graphs: Patterns and a model,” in *SDM*. SIAM, 2007.

[2] E. Aronson, T. D. Wilson, and R. M. Akert, *Social Psychology*, 5th ed. Prentice Hall, Feb. 2004.

[3] J. Brown and P. Reinegen, “Social ties and word-of-mouth referral behavior,” *Journal of Consumer Research*, vol. 14, no. 3, pp. 350–362, 1987.

[4] D. Kempe, J. M. Kleinberg, and É. Tardos, “Maximizing the spread of influence through a social network,” in *KDD*. ACM Press, 2003, pp. 137–146.

[5] M. Cha, J. Antonio, N. Prez, and H. Haddadi, “Flash floods and ripples: The spread of media content through the blogosphere,” in *ICWSM*, 2009.

[6] Y. Liu, X. Huang, A. An, and X. Yu, “Arsa: a sentiment-aware model for predicting sales performance using blogs,” in *SIGIR*. ACM Press, 2007, pp. 607–614.

[7] E. M. Rogers, *Diffusion of innovations*, 5th ed. New York: Free Press, 08 2003.

[8] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl, “Evaluating collaborative filtering recommender systems,” *ACM Trans. Inf. Syst.*, vol. 22, no. 1, pp. 5–53, 2004.

[9] Y. Koren, “Factor in the neighbors: Scalable and accurate collaborative filtering,” *ACM Trans. Knowl. Discov. Data*, vol. 4, no. 1, pp. 1–24, 2010.

[10] J. Leskovec, L. A. Adamic, and B. A. Huberman, “The dynamics of viral marketing,” in *EC*. ACM Press, 2006, pp. 228–237.

[11] J. Huang, X. Cheng, H. Shen, T. Zhou, and X. Jin, “Exploring social influence via posterior effect of word-of-mouth recommendations,” in *WSDM*. ACM Press, 2012, pp. 573–582.

[12] D. M. Romero, W. Galuba, S. Asur, and B. A. Huberman, “Influence and passivity in social media,” in *ECML/PKDD (3)*, 2011, pp. 18–33.

[13] M. Shaw, “Group structure and the behaviour of individuals in small groups,” *Journal of Psychology*, vol. 38, no. 1, pp. 139–149, 1954.

[14] M. Beauchamp, “An improved index of centrality,” *Behavioral Science*, vol. 10, pp. 161–163, 1965.

[15] L. Freeman, “A set of measures of centrality based on betweenness,” *Sociometry*, vol. 40, pp. 35–41, 1977.

[16] R. Geisberger, P. Sanders, and D. Schultes, “Better approximation of betweenness centrality,” in *ALENEX*. SIAM, 2008.

[17] P. Bonacich, “Power and centrality: a family of measures,” *American Journal of Sociology*, vol. 92, pp. 1170–1182, 1987.

[18] G. Pandurangan, P. Raghavan, and E. Upfal, “Using PageRank to Characterize Web Structure,” in *COCOON*. Springer, 2002.

[19] A. Kritikopoulos, M. Sideri, and I. Varlamis, “Blogrank: ranking weblogs based on connectivity and similarity features,” in *AAA-IDEA*. ACM Press, 2006.

[20] N. Agarwal, H. Liu, L. Tang, and P. S. Yu, "Identifying the influential bloggers in a community," in *WSDM*. ACM Press, 2008, pp. 207–218.

[21] J. Zhang, J. Tang, and J. Li, "Expert finding in a social network," in *DASFAA*, 2007, pp. 1066–1069.

[22] J. Zhang, M. S. Ackerman, and L. A. Adamic, "Expertise networks in online communities: structure and algorithms," in *WWW*. ACM Press, 2007, pp. 221–230.

[23] L. Getoor and C. P. Diehl, "Link mining: a survey," *SIGKDD Explorations*, vol. 7, no. 2, pp. 3–12, 2005.

[24] J. Leskovec, D. P. Huttenlocher, and J. M. Kleinberg, "Predicting positive and negative links in online social networks," in *WWW*. ACM Press, 2010, pp. 641–650.

[25] J. Kunegis, A. Lommatzsch, and C. Bauckhage, "The slashdot zoo: mining a social network with negative edges," in *WWW*. ACM Press, 2009, pp. 741–750.

[26] C. Lin and Y. He, "Joint sentiment/topic model for sentiment analysis," in *CIKM*. ACM Press, 2009, pp. 375–384.

[27] D. Liben-Nowell and J. M. Kleinberg, "The link prediction problem for social networks," in *CIKM*, 2003, pp. 556–559.

[28] M. Spiliopoulou, I. Ntoutsi, Y. Theodoridis, and R. Schult, "Monic: modeling and monitoring cluster transitions," in *KDD*. ACM Press, 2006, pp. 706–711.

[29] T. Falkowski, J. Bartelheimer, and M. Spiliopoulou, "Mining and visualizing the evolution of subgroups in social networks," in *Web Intelligence*, 2006, pp. 52–58.

[30] N. Bansal, F. Chiang, N. Koudas, and F. W. Tompa, "Seeking stable clusters in the blogosphere," in *VLDB*. ACM Press, 2007, pp. 806–817.

[31] Y. Matsuo and H. Yamamoto, “Community gravity: measuring bidirectional effects by trust and rating on online social networks,” in *WWW*. ACM Press, 2009, pp. 751–760.

[32] R. Kumar, J. Novak, P. Raghavan, and A. Tomkins, “On the bursty evolution of blogspace,” in *WWW*. ACM Press, 2003, pp. 568–576.

[33] Y.-R. Lin, Y. Chi, S. Zhu, H. Sundaram, and B. L. Tseng, “Analyzing communities and their evolutions in dynamic social networks,” *ACM Trans. Knowl. Discov. Data*, vol. 3, no. 2, pp. 1–31, 2009.

[34] T. Y. Berger-Wolf and J. Saia, “A framework for analysis of dynamic social networks,” in *KDD*. ACM Press, 2006, pp. 523–528.

[35] C. Tantipathananandh, T. Y. Berger-Wolf, and D. Kempe, “A framework for community identification in dynamic social networks,” in *KDD*. ACM Press, 2007, pp. 717–726.

[36] D. Chakrabarti, R. Kumar, and A. Tomkins, “Evolutionary clustering,” in *KDD*. ACM Press, 2006, pp. 554–560.

[37] J. Sun, C. Faloutsos, S. Papadimitriou, and P. S. Yu, “Graphscope: parameter-free mining of large time-evolving graphs,” in *KDD*. ACM Press, 2007, pp. 687–696.

[38] L. Backstrom, D. P. Huttenlocher, J. M. Kleinberg, and X. Lan, “Group formation in large social networks: membership, growth, and evolution,” in *KDD*. ACM Press, 2006, pp. 44–54.

[39] D. J.Watts and S. H.Strogatz, “Collective dynamics of ’small-world’ networks,” *Nature*, vol. 393, pp. 440–442, 1998.

[40] R. Agrawal, S. Rajagopalan, R. Srikant, and Y. Xu, “Mining newsgroups using networks arising from social behavior,” in *WWW*. ACM Press, 2003, pp. 529–535.

[41] M. Cha, A. Mislove, and P. K. Gummadi, “A measurement-driven analysis of information propagation in the flickr social network,” in *WWW*. ACM Press, 2009, pp. 721–730.

[42] W. Chen, Y. Wang, and S. Yang, “Efficient influence maximization in social networks,” in *KDD*. ACM Press, 2009, pp. 199–208.

[43] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. M. VanBriesen, and N. S. Glance, “Cost-effective outbreak detection in networks,” in *KDD*. ACM Press, 2007, pp. 420–429.

[44] W. Chen, C. Wang, and Y. Wang, “Scalable influence maximization for prevalent viral marketing in large-scale social networks,” in *KDD*. ACM Press, 2010, pp. 1029–1038.

[45] H. Li, S. S. Bhowmick, and A. Sun, “Affinity-driven prediction and ranking of products in online product review sites,” in *CIKM*. ACM Press, 2010, pp. 1745–1748.

[46] M. Gomez Rodriguez, J. Leskovec, and A. Krause, “Inferring networks of diffusion and influence,” ser. KDD ’10. ACM Press, 2010, pp. 1019–1028.

[47] S. Borgatti, “The key player problem,” *the National Academy of Sciences Workshop on Terrorism.*, 2002.

[48] H. Bao and E. Y. Chang, “Adheat: an influence-based diffusion model for propagating hints to match ads,” in *WWW*. ACM Press, 2010, pp. 71–80.

[49] E. Bakshy, J. M. Hofman, W. A. Mason, and D. J. Watts, “Everyone’s an influencer: quantifying influence on twitter,” in *WSDM*. ACM Press, 2011, pp. 65–74.

[50] H. Ma, H. Yang, M. R. Lyu, and I. King, “Mining social networks using heat diffusion processes for marketing candidates selection,” in *CIKM*. ACM Press, 2008, pp. 233–242.

[51] A. Pal and S. Counts, “Identifying topical authorities in microblogs,” in *WSDM*. ACM Press, 2011, pp. 45–54.

[52] M. E. Newman, “Clustering and preferential attachment in growing networks,” *Physical Review E*, vol. 64, no. 2, pp. 025 102+, Jul. 2001.

[53] A. L. Barabási and R. Albert, “Emergence of scaling in random networks,” *Science*, vol. 286, no. 5439, pp. 509–512, October 1999.

[54] R. Albert and A. L. Barabási, “Topology of evolving networks: Local events and universality,” *Physical Review Letters*, vol. 85, no. 24, pp. 5234–5237, Dec. 2000.

[55] J. O’Madadhain, J. Hutchins, and P. Smyth, “Prediction and ranking algorithms for event-based network data,” *SIGKDD Explorations*, vol. 7, no. 2, pp. 23–30, Dec. 2005.

[56] K. Cai, S. Bao, Z. Yang, J. Tang, R. Ma, L. Zhang, and Z. Su, “Oolam: an opinion oriented link analysis model for influence persona discovery,” in *WSDM*. ACM Press, 2011, pp. 645–654.

[57] A. Chin and M. H. Chignell, “A social hypertext model for finding community in blogs,” in *Hypertext*. ACM Press, 2006, pp. 11–22.

[58] X. Xu, N. Yuruk, Z. Feng, and T. A. J. Schweiger, “Scan: a structural clustering algorithm for networks,” in *KDD*. ACM Press, 2007, pp. 824–833.

[59] Y. Jo, C. Lagoze, and C. L. Giles, “Detecting research topics via the correlation between graphs and texts,” in *KDD*. ACM Press, 2007, pp. 370–379.

[60] Y.-R. Lin, H. Sundaram, Y. Chi, J. Tatemura, and B. L. Tseng, “Blog community discovery and evolution based on mutual awareness expansion,” in *Web Intelligence*. IEEE Computer Society, 2007, pp. 48–56.

[61] S. Asur, S. Parthasarathy, and D. Ucar, “An event-based framework for characterizing the evolutionary behavior of interaction graphs,” in *KDD*. ACM Press, 2007, pp. 913–921.

[62] H. Li, S. S. Bhowmick, and A. Sun, “Affrank: Affinity-driven ranking of products in online social rating networks,” *Journal of the American Society for Information Science and Technology*, vol. 62, no. 7, pp. 1345–1359, 2011.

[63] J. Leskovec, L. A. Adamic, and B. A. Huberman, “The dynamics of viral marketing,” *ACM Trans. Web*, vol. 1, no. 1, 2007.

[64] D. Pedro and R. Matt, “Mining the network value of customers,” in *KDD*. ACM Press, 2001, pp. 57–66.

[65] S. Asur, S. Parthasarathy, and D. Ucar, “An event-based framework for characterizing the evolutionary behavior of interaction graphs,” *ACM Trans. Knowl. Discov. Data*, vol. 3, no. 4, 2009.

[66] M. Richardson and P. Domingos, “Mining knowledge-sharing sites for viral marketing,” in *KDD*. ACM Press, 2002, pp. 61–70.

[67] G. L. Nemhauser and L. A. Wolsey, “Maximizing submodular set functions: formulations and analysis of algorithms,” *Studies on graphs and discrete programming*, vol. 11, pp. 279–301, 1981.

[68] S. Iwata, “A fully combinatorial algorithm for submodular function minimization.” in *SODA*. SIAM, 2002, pp. 915–919.

[69] S. Iwata and J. B. Orlin, “A simple combinatorial algorithm for submodular function minimization.” in *SODA*. SIAM, 2009, pp. 1230–1237.

[70] U. Feige, V. S. Mirrokni, and J. Vondrak, “Maximizing non-monotone submodular functions,” in *FOCS*. IEEE Computer Society, 2007, pp. 461–471.

[71] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, “An analysis of approximations for maximizing submodular set functions. i,,” *Math. Programming*, vol. 14, no. 3, pp. 265–294, 1978.

[72] Y. Wang, G. Cong, G. Song, and K. Xie, “Community-based greedy algorithm for mining top-k influential nodes in mobile social networks,” in *KDD*. ACM Press, 2010, pp. 1039–1048.

[73] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan, “A data-based approach to social influence maximization,” *Proc. VLDB Endow.*, vol. 5, pp. 73–84, Sep. 2011.

[74] Q. Jiang, G. Song, G. Cong, Y. Wang, W. Si, and K. Xie, “Simulated annealing based influence maximization in social networks,” in *AAAI*, 2011.

[75] W. Chen, Y. Yuan, and L. Zhang, “Scalable influence maximization in social networks under the linear threshold model,” in *ICDM*. IEEE Computer Society, 2010, pp. 88–97.

[76] A. Goyal, W. Lu, and L. V. S. Lakshmanan, “Simpath: An efficient algorithm for influence maximization under the linear threshold model,” in *ICDM*. IEEE Computer Society, 2011, pp. 211–220.

[77] W. Chen, A. Collins, R. Cummings, T. Ke, Z. Liu, D. Rincón, X. Sun, Y. Wang, W. Wei, and Y. Yuan, “Influence maximization in social networks when negative opinions may emerge and propagate,” in *SDM*, 2011, pp. 379–390.

[78] *State of the Blogosphere 2008*, Technorati, http://www.technorati.com/blogging/state-of-the-blogosphere/, 2008.

[79] H. Li, S. S. Bhowmick, and A. Sun, “Blog cascade affinity: analysis and prediction,” in *CIKM*. ACM Press, 2009, pp. 1117–1126.

[80] A. Stewart, L. Chen, R. Paiu, and W. Nejdl, “Discovering information diffusion paths from blogosphere for online advertising,” in *ADKDD*. ACM Press, 2007, pp. 46–54.

[81] D. Gruhl, R. V. Guha, D. Liben-Nowell, and A. Tomkins, “Information diffusion through blogspace,” in *WWW*. ACM Press, 2004, pp. 491–501.

[82] J. Seo and W. B. Croft, “Blog site search using resource selection,” in *CIKM*. ACM Press, 2008, pp. 1053–1062.

[83] N. Bansal and N. Koudas, “Blogscope: spatio-temporal analysis of the blogosphere,” in *WWW*. ACM Press, 2007, pp. 1269–1270.

[84] D. Watts, “A simple model of global cascades on random networks,” *P Natl Acad Sci USA*, vol. 99, no. 9, pp. 5766–5771, 2002.

[85] S. Bikhchandani, D. Hirshleifer, and I. Welch, “A theory of fads, fashion, custom, and cultural change as informational cascades,” *The Journal of Political Economy*, vol. 100, no. 5, pp. 992–1026, 1992.

[86] X. Shi, J. Zhu, R. Cai, and L. Zhang, “User grouping behavior in online forums,” in *KDD*. ACM Press, 2009, pp. 777–786.

[87] J. Leskovec, L. Backstrom, R. Kumar, and A. Tomkins, “Microscopic evolution of social networks,” in *KDD*. ACM Press, 2008, pp. 462–470.

[88] M. McGlohon, J. Leskovec, C. Faloutsos, M. Hurst, and N. Glance, “Finding patterns in blog shapes and blog evolution,” in *ICWSM*, 2007.

[89] M. E. J. Newman, “The structure and function of complex networks,” *SIAM Review*, vol. 45, no. 2, pp. 167–256, 2003.

[90] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[91] D. C. Liu and J. Nocedal, “On the limited memory BFGS method for large scale optimization,” *Mathematical Programming*, vol. 45, no. 3, pp. 503–528, 1989.

[92] H. Li, S. S. Bhowmick, and A. Sun, “Casino: towards conformity-aware social influence analysis in online social networks,” in *CIKM*. ACM Press, 2011, pp. 1007–1012.

[93] J. Leskovec, D. P. Huttenlocher, and J. M. Kleinberg, “Signed networks in social media,” in *CHI*. ACM Press, 2010, pp. 1361–1370.

[94] S. A. Munson and P. Resnick, “Presenting diverse political opinions: how and how much,” in *CHI*. ACM Press, 2010, pp. 1457–1466.

[95] C. Herbert, “Compliance, identification, and internalization: Three processes of attitude change,” *The Journal of Conflict Resolution*, vol. 2, no. 1, 1958.

[96] R. S. Baron, J. A. Vandello, and B. Brunsman, “The forgotten variable in conformity research: Impact of task importance on social influence,” *Journal of Personality and Social Psychology*, vol. 71, pp. 915–927, 1996.

[97] B. H.Hodges and A. L.Geyer, “A nonconformist account of the asch experiments: Values, pragmatics, and moral dilemmas,” *Personality and Social Psychology Review*, vol. 10, no. 1, pp. 2–19, 2006.

[98] D. J. Watts and P. S. Dodds., “Influentials, networks, and public opinion formation,” *Journal of Consumer Research*, vol. 34, pp. 441–458, 2007.

[99] A-lias. (2008) Lingpipe 4.0.1. http://alias-i.com/lingpipe.

[100] Y. Jo and A. H. Oh, "Aspect and sentiment unification model for online review analysis," in *WSDM*. ACM Press, 2011, pp. 815–824.

[101] R. K. Pon, A. F. Cardenas, D. Buttler, and T. Critchlow, "Tracking multiple topics for finding interesting articles," in *KDD*. ACM Press, 2007, pp. 560–569.

[102] X. Yu, Y. Liu, X. Huang, and A. An, "A quality-aware model for sales prediction using reviews," in *WWW*. ACM Press, 2010, pp. 1217–1218.

[103] G. H. Golub and C. F. V. Loan, *Matrix Computations*, 3rd ed. The Johns Hopkins University Press, 1996.

[104] T. Joachims, "Making large-scale SVM learning practical," in *Advances in Kernel Methods - Support Vector Learning*, B. Schölkopf, C. Burges, and A. Smola, Eds. MIT Press, Cambridge, MA, USA, 1999.

[105] H. Fock, A. K. Chan, and D. Yan, "Member-organization connection impacts in affinity marketing," *Journal of Business Research*, vol. 64, no. 7, pp. 672–679, 2010.

[106] Q. Feng, K. Hwang, and Y. Dai, "Rainbow product ranking for upgrading e-commerce," *IEEE Internet Computing*, vol. 13, no. 5, pp. 72–80, 2009.

[107] X. Li, L. Guo, and Y. E. Zhao, "Tag-based social interest discovery," in *WWW*. ACM Press, 2008, pp. 675–684.

[108] H. T. Shen, B. C. Ooi, and X. Zhou, "Towards effective indexing for very large video sequence database," in *SIGMOD*. ACM Press, 2005, pp. 730–741.

[109] P. Singla and M. Richardson, "Yes, there is a correlation: - from social networks to personal behavior on the web," in *WWW*. ACM Press, 2008, pp. 655–664.

[110] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos, “A unified framework for providing recommendations in social tagging systems based on ternary semantic analysis,” *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 2, pp. 179–192, 2010.

[111] M. Clements, A. P. De Vries, and M. J. T. Reinders, “The task-dependent effect of tags and ratings on social media access,” *ACM Trans. Inf. Syst.*, vol. 28, no. 21, pp. 1–42, 2010.

[112] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*. Springer, Morgan Kaufmann Publishers Inc. 2005.

[113] P. Holme, C. R. Edling, and F. Liljeros, “Structure and time evolution of an internet dating community,” *Social Networks*, vol. 26, no. 2, pp. 155–174, May 2004.

[114] T. Opsahl and P. Panzarasa, “Clustering in weighted networks,” *Social Networks*, vol. 31, no. 2, pp. 155–163, May 2009.

[115] C. Chen, F. Ibekwe-Sanjuan, and J. Hou, “The structure and dynamics of cocitation clusters: A multiple-perspective cocitation analysis.” *J. Am. Soc. Inf. Sci. Technol.*, vol. 61, no. 7, pp. 1386–1409, 2010.

[116] M. de Klepper, E. Sleebos, G. van de Bunt, and F. Agneessens, “Similarity in friendship networks: Selection or influence? the effect of constraining contexts and non-visible individual attributes,” *Social Networks*, August 2009.

[117] J. Tang and J. Zhang, “Modeling the evolution of associated data,” *Data Knowl. Eng.*, vol. 69, no. 9, pp. 965–978, 2010.

[118] H. Sakoe and S. Chiba, “Dynamic programming algorithm optimization for spoken word recognition,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, no. 1, pp. 43–49, 1978.

[119] S. J. Pan, X. Ni, J.-T. Sun, Q. Yang, and Z. Chen, "Cross-domain sentiment classification via spectral feature alignment," in *WWW*. ACM Press, 2010, pp. 751–760.

[120] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Foundations and Trends in Information Retrieval*, vol. 2, no. 1–2, pp. 1–135, 2008.

[121] P. J. Brockwell and R. A. Davis, *Introduction to Time Series and Forecasting*. Springer, March 2002.

[122] H. Akaike, "A new look at the statistical model identification," *IEEE Trans. Aut. Contr.*, vol. 19, pp. 716–723, 1974.

[123] K. Järvelin and J. Kekäläinen, "Cumulated gain-based evaluation of ir techniques," *ACM Trans. Inf. Syst.*, vol. 20, no. 4, pp. 422–446, 2002.

[124] D. Agarwal, E. Gabrilovich, R. Hall, V. Josifovski, and R. Khanna, "Translating relevance scores to probabilities for contextual advertising," in *CIKM*. ACM Press, 2009, pp. 1899–1902.

[125] J. Coleman, H. Menzel, and E. Katz, *Medical Innovations: A Diffusion Study*. Bobbs Merrill, 1966.

[126] T. Valente, *Network Models of the Diffusion of Innovations*. Hampton Press, 1995.

[127] V. Mahajan, E. Muller, and F. Bass, "New product diffusion models in marketing: A review and directions for research," *Journal of Marketing*, vol. 54, no. 1, pp. 1–26, 1990.

[128] J. Goldenberg, B. Libai, and E. Muller, "Talk of the network: A complex systems look at the underlying process of word-of-mouth," *Marketing Letters*, vol. 12, no. 3, pp. 211–223, 2001.

[129] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," in *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. 12, June 2002, pp. 7821–7826.

[130] R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins, "Trawling the Web for emerging cyber-communities," *Computer Networks*, vol. 31, no. 11–16, pp. 1481–1493, 1999.

[131] M. McGlohon, L. Akoglu, and C. Faloutsos, "Weighted graphs and disconnected components: patterns and a generator," in *KDD*, 2008, pp. 524–532.

[132] A. Clauset, "Finding local community structure in networks," *Physical Review E*, vol. 72, no. 2, p. 026132, 2005.

[133] A. Clauset, C. Moore, and M. E. J. Newman, "Finding community structure in very large networks," *Physical Review E*, vol. 70, no. 6, p. 066111, 2004.

[134] M. E. Newman, "Modularity and community structure in networks." in *Proceedings of the National Academy of Sciences of the United States of America*, vol. 103, no. 23, June 2006, pp. 8577–8582.

[135] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E*, vol. 69, no. 2, p. 026113, 2004.

[136] K. Schloegel, G. Karypis, and V. Kumar, "Parallel static and dynamic multi-constraint graph partitioning." *Concurrency and Computation: Practice and Experience*, vol. 14, no. 3, pp. 219–240, 2002.

[137] N. Wang, S. Parthasarathy, K.-L. Tan, and A. K. H. Tung, "Csv: visualizing and mining cohesive subgraphs," in *SIGMOD*. ACM Press, 2008, pp. 445–458.

[138] T. Condie, N. Conway, P. Alvaro, J. M. Hellerstein, J. Gerth, J. Talbot, K. Elmeleegy, and R. Sears, “Online aggregation and continuous query support in mapreduce,” in *SIGMOD*. ACM Press, 2010, pp. 1115–1118.

[139] S. Das, Y. Sismanis, K. S. Beyer, R. Gemulla, P. J. Haas, and J. McPherson, “Ricardo: integrating r and hadoop,” in *SIGMOD*. ACM Press, 2010, pp. 987–998.

[140] A. Thusoo, Z. Shao, S. Anthony, D. Borthakur, N. Jain, J. Sen Sarma, R. Murthy, and H. Liu, “Data warehousing and analytics infrastructure at facebook,” in *SIGMOD*. ACM Press, 2010, pp. 1013–1020.

[141] Y. Xu, P. Kostamaa, and L. Gao, “Integrating hadoop and parallel dbms,” in *SIGMOD*. ACM Press, 2010, pp. 969–974.

[142] F. Pellegrini and J. Roman, “Scotch: A software package for static mapping by dual recursive bipartitioning of process and architecture graphs,” in *HPCN*. Brussels, Belgium. LNCS 1067: Springer, April 1996, pp. 493–498.

[143] B. Strauss, “Social facilitation in motor tasks: a review of research and theory,” *Psychology of Sport and Exercise*, vol. 3, pp. 237–256, 2001.

[144] D. Myers and H. Lamm, “The polarizing effect of group discussion,” *American Scientist*, vol. 63, no. 3, pp. 297–303, 1975.

Appendix

List of Publications

- **H. Li**, S. S Bhowmick, and A. Sun. *AffRank*: Affinity-Driven Ranking of Products in Online Social Rating Networks. *Journal of the American Society for Information Science and Technology 62(7), pages 1345-1359*, 2011. WILEY Press.
- **H. Li**, S. S Bhowmick, and A. Sun. CASINO: Towards Conformity-aware Social Influence Analysis in Online Social Networks. In *CIKM'11: Proceeding of the 20th ACM conference on Information and Knowledge Management*, 2011.
- **H. Li**, S. S Bhowmick, and A. Sun. Affinity-Driven Prediction and Ranking of Products in Online Product Review Sites. In *CIKM'10: Proceeding of the 19th ACM conference on Information and Knowledge Management*, pages 1745–1748, 2010.
- **H. Li**, S. S Bhowmick, and A. Sun. Blog cascade affinity: analysis and prediction. In *CIKM'09: Proceeding of the 18th ACM conference on Information and Knowledge Management*, pages 1117–1126, 2009.
- **H. Li**, S. S Bhowmick, and A. Sun. PATINA: A Partitioning-based Efficient Strategy for Influence Maximization in Large Online Social Networks. *Submitted for publication.*
- **H. Li**, S. S Bhowmick, and A. Sun. CINEMA: Towards Conformity-Aware Influence Maximization in Large Online Social Networks. *Submitted for publication.*
- **H. Li**, S. S Bhowmick, and A. Sun. Blog cascade affinity: analysis and prediction. *Submitted for publication.*